Operating System/2™ Programming Tools and Information Version 1.2

Presentation Manager C/2 Bindings Reference

64F0278

Operating System/2™
Programming Tools
and Information
Version 1.2

Presentation Manager C/2 Bindings Reference

First Edition (September 1989)

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.

It is possible that this publication may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used.

Requests for technical information about IBM products should be made to your IBM Authorized Dealer or your IBM Marketing Representative.

Note to US Government users — Documentation related to Restricted Rights — Use, duplication, or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Special Notices

The following names, used in this publication, are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries:

IBM
Operating System/2
OS/2
Presentation Manager
Systems Application Architecture.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license enquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Armonk, NY 10504.

Library Structure

OS/2 Product

IBM Operating System/2
Standard Edition
Version 1.2

Getting Started
Using Advanced Features
Product Information

6024926 3.5″ diskette
6024930 5.25″ diskette

OS/2 Programming Tools and Information

IBM Operating System/2
Version 1.2

Installation
Programming Overview
Programming Guide
Building Programs
Dialog Tag Language Guide and Reference
Dialog Manager Guide and Reference
Dialog Manager and Dialog Tag Language Reference Summary
Programming Reference (3 books)
Bindings Reference for Presentation Manager (4 books for COBOL/2, FORTRAN/2, C/2, and Macro Assembler/2)
I/O Subsystems and Device Support (2 books)
Systems Application Architecture
Common User Access: Advanced Interface Design Guide

6024929

Separate Order (no charge)

Keyboards and Code Pages

6280345

Available Separately

Command Reference

6024928

Service Coordinator's Guide

15F2214

Programming Languages

IBM Basic Compiler/2
Version 1.0 6280179

IBM Macro Assembler/2
Version 1.0 6280181

IBM Pascal Compiler/2
Version 1.0 6280183

IBM FORTRAN/2
Version 1.02 6280185

IBM COBOL/2
Version 1.0 6280207

IBM C/2
Version 1.1 6280284

Systems Application Architecture

An Overview

GC26-4341

Writing Applications: A Design Guide

SC26-4362

Common User Access: Advanced Interface Design Guide
SC26-4582

Common User Access: Basic Interface Design Guide
SC26-4583

Common Programming Interface

C Reference - Level 2

SC09-1308

COBOL Reference

SC26-4354

Dialog Reference

SC26-4356

FORTRAN Reference

SC26-4357

Procedures Language Reference
SC26-4358

Presentation Reference

SC26-4359

Contents

Chapter 1. Introduction

This book describes the Operating System/2 (OS/2) Version 1.2 language binding for IBM C/2.

It provides language-dependent information about the functions described in the OS/2 Version 1.2 *Presentation Manager Programming Reference*.

The information in this book, together with the functional descriptions detailed in the *Presentation Manager Programming Reference*, enable the user to generate call statements in C.

The following information is provided:

- The syntax of each data type and structure
- The parameter list for each call.

Header Files

All functions require an "include" for the system header file OS2.H:

```
#include  <OS2.H>
```

Also, most functions require a "define" to select an appropriate (conditional) section of the header file, and hence, the required entry point. Where this is necessary, it is shown at the head of the function definition in the form:

```
#define   INCL_name
```

Programming Note: These "#defines" must precede the "#include <OS2.H>."

APIENTRY

Unless shown otherwise, all functions have an implicit type of APIENTRY, which is defined in the system header file OS2.H as:

```
#define APIENTRY pascal far
```

Miscellaneous Definitions

The following definitions are provided in the system header file OS2.H:

```
#define FALSE    0
#define TRUE     1
#define NULL     0
#define FAR      far
#define NEAR     near
#define PASCAL   pascal
```

IBM C/2 Data Types

If you are writing an application in IBM C/2, the Pascal calling conventions must be used. IBM C/2, unlike other high-level languages, pushes the parameters for each subroutine from right to left onto a stack. The calling application rather than the called subroutine must remove the arguments from the stack.

In the case of messages, the parameters **param1**, **param2**, and **reply** can be a combination of several pieces of information. These parameters contain one of the following:

- A simple data type, that can be constructed or analyzed by means of C data declaration facilities
- A combination of simple data types, that can be constructed or analyzed by means of C data declaration facilities
- A pointer to a compound data type, that can be constructed or analyzed by means of the corresponding compound data type.

Helper Macros

A series of macros is defined for packing data into, and extracting data from, variables of *MPARAM* and *MRESULT* data types. They are used in conjunction with the WinSendMsg and the other message functions, and also inside window and dialog procedures.

These macros always cast their arguments to the specified type, so values of any of the types specified for each macro may be passed without additional casting. NULL may be used to pass unused parameter data.

Macros for packing data into a *MPARAM* variable:

```
/* Used to pass any pointer type: */
#define MPFROMP(p)              ((MPARAM)(VOID FAR *)(p))

/* Used to pass a window handle: */
#define MPFROMHWND(hwnd)        ((MPARAM)(HWND)(hwnd))

/* Used to pass a CHAR, UCHAR, or BYTE: */
#define MPFROMCHAR(ch)          ((MPARAM)(USHORT)(ch))

/* Used to pass a SHORT, USHORT, or BOOL: */
#define MPFROMSHORT(s)          ((MPARAM)(USHORT)(s))

/* Used to pass two SHORTs, USHORTs, or BOOLs: */
#define MPFROM2SHORT(s1, s2)  ((MPARAM)MAKELONG(s1, s2))

/* Used to pass a SHORT and 2 UCHARs: (WM_CHAR msg)*/
#define MPFROMSH2CH(s, uch1, uch2)
 ((MPARAM)MAKELONG(s, MAKESHORT(uch1, uch2)))

/* Used to pass a LONG or ULONG: */
#define MPFROMLONG(l)           ((MPARAM)(ULONG)(l))
```

Macros for extracting data from a *MPARAM* variable:

```
/* Used to get any pointer type: */
#define PVOIDFROMMP(mp)      ((VOID FAR *)(mp))

/* Used to get a window handle: */
#define HWNDFROMMP(mp)       ((HWND)(mp))
```

```
/* Used to get CHAR, UCHAR, or BYTE: */
#define CHAR1FROMMP(mp)      ((UCHAR)(mp))
#define CHAR2FROMMP(mp)      ((UCHAR)((ULONG)mp >> 8))
#define CHAR3FROMMP(mp)      ((UCHAR)((ULONG)mp >> 16))
#define CHAR4FROMMP(mp)      ((UCHAR)((ULONG)mp >> 24))

/* Used to get a SHORT, USHORT, or BOOL: */
#define SHORT1FROMMP(mp)     ((USHORT)(ULONG)(mp))
#define SHORT2FROMMP(mp)     ((USHORT)((ULONG)mp >> 16))

/* Used to get a LONG or ULONG: */
#define LONGFROMMP(mp)       ((ULONG)(mp))
```

Macros for packing data into a *MRESULT* variable:

```
/* Used to pass any pointer type: */
#define MRFROMP(p)             ((MRESULT)(VOID FAR *)(p))

/* Used to pass a SHORT, USHORT, or BOOL: */
#define MRFROMSHORT(s)         ((MRESULT)(USHORT)(s))

/* Used to pass two SHORTs, USHORTs, or BOOLs: */
#define MRFROM2SHORT(s1, s2)  ((MRESULT)MAKELONG(s1, s2))

/* Used to pass a LONG or ULONG: */
#define MRFROMLONG(l)          ((MRESULT)(ULONG)(l))
```

Macros for extracting data from a *MRESULT* variable:

```
/* Used to get any pointer type: */
#define PVOIDFROMMR(mr)      ((VOID FAR *)(mr))

/* Used to get a SHORT, USHORT, or BOOL: */
#define SHORT1FROMMR(mr)     ((USHORT)((ULONG)mr))
#define SHORT2FROMMR(mr)     ((USHORT)((ULONG)mr >> 16))

/* Used to get a LONG or ULONG: */
#define LONGFROMMR(mr)       ((ULONG)(mr))
```

The following macros are for use with *DDESTRUCT* and *DDEINIT* structures:

```
/* Used to return a PSZ pointing to the DDE item name: */
#define DDES_PSZITEMNAME(pddes) \
        (((PSZ)pddes) + ((PDDESTRUCT)pddes)->offszItemName)

/* Used to return a PBYTE pointing to the DDE data: */
#define DDES_PABDATA(pddes)       \
        (((PBYTE)pddes) + ((PDDESTRUCT)pddes)->offabData)

/* Used to convert a selector to a PDDESTRUCT: */
#define SELTOPDDES(sel)              ((PDDESTRUCT)MAKEP(sel, 0))

/* Used to PDDESTRUCT to a selector for freeing / reallocating: */
#define PDDESTOSEL(pddes)            (SELECTOROF(pddes))

/* Used to PDDEINIT to a selector for freeing: */
#define PDDEITOSEL(pddei)            (SELECTOROF(pddei))
```

The following macros are for common operations on control windows:

```
/*Used to set the check state of a button */
#define WinCheckButton(hwndDlg, id, usCheckState) \
    ((USHORT)WinSendDlgItemMsg(hwndDlg, id, BM_SETCHECK, \
             MPFROMSHORT(usCheckState), (MPARAM)NULL))

/*Used to query the check state of a button */
#define WinQueryButtonCheckstate(hwndDlg, id) \
    ((USHORT)WinSendDlgItemMsg(hwndDlg, id, BM_QUERYCHECK, \
            MPARAM(NULL), (MPARAM)NULL))

/*Used to enable/disable a control in a dialog */
#define WinEnableControl(hwndDlg, id, fEnable) \
    WinEnableWindow(WinWindowFromID(hwndDlg, id), fEnable)

/*Used to check the enabled state of a control in a dialog */
#define WinIsControlEnabled(hwndDlg, id) \
    ((BOOL)WinIsWindowEnabled(WinWindowFromID(hwndDlg, id)))

/*Used to delete a listbox item                */
#define WinDeleteLboxItem(hwndLbox, index) \
    ((SHORT)WinSendMsg(hwndLbox, LM_DELETEITEM, MPFROMSHORT(index), \
            (MAPRAM)NULL))

/*Used to insert an item into a listbox        */
#define WinInsertLboxItem(hwndLbox, index, psz) \
    ((SHORT)WinSendMsg(hwndLboc, LM_INSERTITEM, MPFROMSHORT(index), \
            MPFROMP(psz)))

/*Used to query the number of items in a listbox */
#define WinQueryLboxCount(hwndLbox) \
    ((SHORT)WinSendMsg(hwndLbox, LM_QUERYITEMCOUNT, (MPARAM)NULL, \
            (MPARAM)NULL))

/*Used to query the text of a listbox item     */
#define WinQueryLboxItemText(hwndLbox, index, psz, cchMax) \
    ((SHORT)WinSendMsg(hwndLbox, LM_QUERYITEMTEXT, \
            MPFROM2SHORT((index), (cchMax)), MPFROMP(psz)))

/*Used to query the text length of a listbox item */
#define WinQueryLboxItemTextLength(hwndLbox, index)         \
    ((SHORT)WinSendMsg(hwndLbox, LM_QUERYITEMTEXTLENGTH,    \
            MPFROMSHORT(index), (MPARAM)NULL))

/*Used to set the text of a listbox item       */
#define WinSetLboxItemText(hwndLbox, index, psz) \
    ((BOOL)WinSendMsg(hwndLbox, LM_SETITEMTEXT, \
            MPFROMSHORT(index), MPFROMP(psz)))

/*Used to query text of the selected listbox item */
/*(Single selection listboxes only)              */
 #define WinQueryLboxSelectedItem(handLbox) \
     ((SHORT)WinSendMsg(hwndLbox, LM_QUERYSELECTION, MPFROMSHORT(LIT_FIRST), \
             (MPARAM)NULL))
```

Addressing Elements in Arrays

Constants defining array elements are given values that are zero-based in C; that is, the numbering of the array elements starts at zero, not one.

For example, in the DevQueryCaps function, the sixth element of the **Array** parameter is CAPS_HEIGHT, which is equated to **5**.

Count parameters related to such arrays always mean the actual number of elements available. Therefore, again using the DevQueryCaps function as an example, if all elements up to and including CAPS_HEIGHT are provided for, **Count** could be set to (CAPS_HEIGHT + 1).

In functions for which the starting array element can be specified, this is always zero-based, and so the C element number constants can be used directly. For example, to start with the CAPS_HEIGHT element, the **Start** parameter can be set to CAPS_HEIGHT.

Implicit Pointer Data Types

In this binding, a data type name beginning with "P" (for example, PERRORCODE) is likely to be a pointer to another data type (in this instance, ERRORCODE).

In the following data type summary no explicit "typedefs" are shown for pointers.

Therefore, if no data type definition can be found in the summary for a data type name "Pxxxxxx," it becomes a pointer to the data type "xxxxxx," for which a definition should be found in the summary.

The implicit type definition needed for such a pointer "Pxxxxxx" is:

```
typedef xxxxxx FAR *Pxxxxxx;
```

Such definitions are provided by means of the system header file OS2.H.

Storage Mapping of Data Types

The storage mapping of the data types is dependant on the machine architecture. In order to be portable, applications must access the data types using the definitions supplied for that environment.

Mapping Metalanguage Data Types to C Data Types

Some data types shown in Chapter 2 of this book do not occur in C, but belong to the metalanguage data types described in Chapter 2 of the *Presentation Manager Programming Reference*. They are included to provide a mapping to the corresponding C data types, and can be identified as those that contain the phrase "(Maps via...)."

This treatment is not needed for every metalanguage data type, as many of them have the same name as the corresponding C data type.

Chapter 2. Data Types

ACCEL

Accelerator structure.

```
typedef struct _ACCEL {
  USHORT    fs;   /*  Options */
  USHORT    key;  /*  Key */
  USHORT    cmd;  /*  Command code */
 } ACCEL;
```

ACCELTABLE

Accelerator-table structure.

```
typedef struct _ACCELTABLE {
  USHORT    cAccel;     /*   Number of accelerator entries */
  USHORT    codepage;   /*   Code page for accelerator entries */
  ACCEL     aaccel[1];  /*   Accelerator entries */
 } ACCELTABLE;
```

ARCPARAM

Arc-parameters structure.
(Maps via ARCPARAMS)

ARCPARAMS

Arc-parameters structure.

```
typedef struct _ARCPARAMS {
  LONG   lP;   /*  P coefficient */
  LONG   lQ;   /*  Q coefficient */
  LONG   lR;   /*  R coefficient */
  LONG   lS;   /*  S coefficient */
 } ARCPARAMS;
```

AREABUNDLE

Area-attributes bundle structure.

```
typedef struct _AREABUNDLE {
  LONG     lColor;           /* Area foreground color */
  LONG     lBackColor;       /* Area background color */
  USHORT   usMixMode;        /* Area foreground-mix mode */
  USHORT   usBackMixMode;    /* Area background-mix mode */
  USHORT   usSet;            /* Pattern set */
  USHORT   usSymbol;         /* Pattern symbol */
  POINTL   ptlRefPoint;      /* Pattern reference point */
 } AREABUNDLE;
```

ATOM

Atom identity.

```
typedef USHORT ATOM;
```

BANDRECT

Rectangle structure, used for the coordinates of an output band (see DevEscape).

An empty rectangle is one for which **xleft** is greater than **xright**, or **ybottom** is greater than **ytop**.

```
typedef struct _BANDRECT {
  LONG   lxLeft;     /*  x coordinate of left edge of rectangle */
  LONG   lyBottom;   /*  y coordinate of bottom edge of rectangle
                         */
  LONG   lxRight;    /*  x coordinate of right edge of rectangle
                         */
  LONG   lyTop;      /*  y coordinate of top edge of rectangle */
 } BANDRECT;
```

BITMAPINFO Bit-map information structure.

Each bit plane logically contains (**bitmapwidth*bitmapheight*bitcount**) bits, although the actual length can be greater because of padding.

```
typedef struct _BITMAPINFO {
  ULONG    cbFix;          /*  Length of fixed portion of structure
                               */
  USHORT   cx;             /*  Bit-map width in pels */
  USHORT   cy;             /*  Bit-map height in pels */
  USHORT   cPlanes;        /*  Number of bit planes */
  USHORT   cBitCount;      /*  Number of bits per pel within a
                              plane */
  RGB      argbColor[1];   /*  Array of RGB values */
} BITMAPINFO;
```

BITMAPINFOHEADER Bit-map information header structure.

Each bit plane logically contains (**bitmapwidth*bitmapheight*bitcount**) bits, although the actual length can be greater because of padding.

```
typedef struct _BITMAPINFOHEADER {
  ULONG    cbFix;        /*  Length of structure */
  USHORT   cx;           /*  Bit-map width in pels */
  USHORT   cy;           /*  Bit-map height in pels */
  USHORT   cPlanes;      /*  Number of bit planes */
  USHORT   cBitCount;    /*  Number of bits per pel within a plane
                            */
} BITMAPINFOHEADER;
```

BIT8 Defines eight independent BOOL values.
(Maps via UCHAR)

BIT16 Defines 16 independent BOOL values.
(Maps via USHORT)

BIT32 Defines 32 independent BOOL values.
(Maps via ULONG)

BOOL Boolean.

The *BOOL* data type is defined such that a nonzero value indicates TRUE, and a zero value indicates FALSE.

The constants TRUE and FALSE have been defined as 1 and 0 respectively. The constant TRUE should never be used in an equality comparison with a *BOOL* as this may result in incorrect results.

When an application needs to pass a TRUE value on a function call, or as a return value from a window procedure, it is recommended that it passes the value TRUE, which is 1, and not a nonzero value.

```
typedef unsigned short BOOL;
```

BTNCDATA Button-control data structure.

```
typedef struct _BTNCDATA {
  USHORT    cb;            /*    Length of the control data in
                              bytes */
  USHORT    fsCheckState;  /*    Check state of button */
  USHORT    fsHiliteState; /*    Highlighting state of button */
} BTNCDATA;
```

BUFFER Buffer.
(Maps via PBYTE)

BUNDLE

Bundle data area. It is overlaid by one of these attribute bundle structures:

AREABUNDLE
CHARBUNDLE
IMAGEBUNDLE
LINEBUNDLE
MARKERBUNDLE.

(Maps via PBUNDLE)

BYTE

Byte.

```
typedef unsigned char BYTE;
```

CATCHBUF

Saved execution environment buffer.

```
typedef struct _CATCHBUF {
  ULONG    reserved[4];  /*   Save area */
 } CATCHBUF;
```

CHAR

Single-byte character.

```
#define CHAR char
```

CHARBUNDLE

Character-attributes bundle structure.

```
typedef struct _CHARBUNDLE {
  LONG     lColor;            /* Character foreground color */
  LONG     lBackColor;        /* Character background color */
  USHORT   usMixMode;         /* Character foreground-mix mode */
  USHORT   usBackMixMode;     /* Character background-mix mode */
  USHORT   usSet;             /* Character set */
  USHORT   usPrecision;       /* Character precision */
  SIZEF    sizfxCell;         /* Character cell size */
  POINTL   ptlAngle;          /* Character angle */
  POINTL   ptlShear;          /* Character shear */
  USHORT   usDirection;       /* Character direction */
 } CHARBUNDLE;
```

CLASSINFO

Class-information structure.

```
typedef struct _CLASSINFO {
  ULONG    flClassStyle;  /*     Class-style flags */
  PFNWP    pfnWindowProc;  /*    Window procedure */
  USHORT   cbWindowData;  /*     Number of additional window words
                                 */
 } CLASSINFO;
```

COUNT2

Count in the range 0 through 65 535.
(Maps via USHORT)

COUNT2B

Count of bytes, in the range 0 through 65 535.
(Maps via USHORT)

COUNT2CH

Count of characters, in the range 0 through 65 535.
(Maps via USHORT)

COUNT4

Count in the range 0 through 4 294 967 295.
(Maps via ULONG)

COUNT4B

Count of bytes, in the range 0 through 4 294 967 295.
(Maps via ULONG)

CPID

Code-page identity.
(Maps via USHORT)

CREATEPARAMS

Create parameters structure.
(Maps via STORAGE)

CREATESTRUCT Create-window data structure.

```
typedef struct _CREATESTRUCT {
  PVOID   pPresParams;        /*   Presentation parameters */
  PVOID   pCtlData;           /*   Control data */
  USHORT  id;                 /*   Window identifier */
  HWND    hwndInsertBehind;   /*   Window behind which the window
                                   is to be placed */
  HWND    hwndOwner;          /*   Window owner */
  SHORT   cy;                 /*   Window height */
  SHORT   cx;                 /*   Window width */
  SHORT   y;                  /*   y coordinate of origin */
  SHORT   x;                  /*   x coordinate of origin */
  ULONG   flStyle;            /*   Window style */
  PSZ     pszText;            /*   Window text */
  PSZ     pszClass;           /*   Registered window class name */
  HWND    hwndParent;         /*   Parent window handle */
 } CREATESTRUCT;
```

CTLDATA Class-specific control data, beginning with a value conforming to a *COUNT2B* data type, which specifies the overall length of the data.

The following are cases of this data type:

BTNCDATA	Button control data.
ENTRYFDATA	Entry field control data.
FRAMECDATA	Frame control data.
SBCDATA	Scroll bar control data.
MLECDATA	Multi-line entry field control data.

(Maps via PVOID)

CURSORINFO Cursor-information structure.

```
typedef struct _CURSORINFO {
  HWND    hwnd;     /*  Window handle */
  SHORT   x;        /*  x coordinate */
  SHORT   y;        /*  y coordinate */
  SHORT   cx;       /*  Cursor width */
  SHORT   cy;       /*  Cursor height */
  USHORT  fs;       /*  Options */
  RECTL   rclClip;  /*  Cursor box */
 } CURSORINFO;
```

DDEINIT Dynamic data exchange initiation structure.

```
typedef struct _DDEINIT {
  USHORT   cb;            /* Length of structure */
  PSZ      pszAppName;    /* Application name */
  PSZ      pszTopic;      /* Topic */
 } DDEINIT;
```

DDESTRUCT Dynamic data exchange control structure.

```
typedef struct _DDESTRUCT {
  ULONG   cbData;          /*  Total length */
  USHORT  fsStatus;        /*  Status */
  USHORT  usFormat;        /*  Data format */
  USHORT  offszItemName;   /*  Offset to item */
  USHORT  offabData;       /*  Offset to beginning of data */
 } DDESTRUCT;
```

DEVOPENDATA Open-device data area. The format of this area is the same as a *DEVOPENSTRUC* structure.
(Maps via PDEVOPENDATA)

DEVOPENSTRUC Open-device data structure.

The same structure is applicable to both DevOpenDC and SplQmOpen calls.

```
typedef struct _DEVOPENSTRUC {
  PSZ         pszLogAddress;          /* Logical address */
  PSZ         pszDriverName;          /* Driver name */
  PDRIVDATA   pdriv;                  /* Driver data */
  PSZ         pszDataType;            /* Data type */
  PSZ         pszComment;             /* Comment */
  PSZ         pszQueueProcName;       /* Queue-processor name */
  PSZ         pszQueueProcParams;     /* Queue-processor
                                         parameters */
  PSZ         pszSpoolerParams;       /* Spooler parameters */
  PSZ         pszNetworkParams;       /* Network parameters */
 } DEVOPENSTRUC;
```

DLGTEMPLATE Dialog-template structure.

```
typedef struct _DLGTEMPLATE {
  USHORT     cbTemplate;         /*   Length of template */
  USHORT     type;               /*   Template format type */
  USHORT     codepage;           /*   Code page */
  USHORT     offadlgti;          /*   Offset to dialog items */
  USHORT     fsTemplateStatus;   /*   Template status */
  USHORT     iItemFocus;         /*   Index of item to receive
                                  focus initially */
  USHORT     coffPresParams;     /*   Count of
                                  presentation-parameter offsets
                                  */
  DLGTITEM   adlgti[1];          /*   Start of dialog items */
 } DLGTEMPLATE;
```

DLGTITEM Dialog-item structure.

```
typedef struct _DLGTITEM {
  USHORT    fsItemStatus;   /*    Status */
  USHORT    cChildren;      /*    Count of children to this dialog
                            item */
  USHORT    cchClassName;   /*    Length of class name */
  USHORT    offClassName;   /*    Offset to class name */
  USHORT    cchText;        /*    Length of text */
  USHORT    offText;        /*    Offset to text */
  ULONG     flStyle;        /*    Dialog item window style */
  SHORT     x;              /*    x coordinate of origin of
                            dialog-item window */
  SHORT     y;              /*    y coordinate of origin of
                            dialog-item window */
  SHORT     cx;             /*    Dialog-item window width */
  SHORT     cy;             /*    Dialog-item window height */
  USHORT    id;             /*    Identity */
  USHORT    offPresParams;  /*    Reserved */
  USHORT    offCtlData;     /*    Offset to control data */
 } DLGITEM;
```

DRIVDATA

Driver-data structure.

```
typedef struct _DRIVDATA {
  LONG   cb;                /*  Length */
  LONG   lVersion;          /*  Version */
  CHAR   szDeviceName[32];  /*  Device name */
  CHAR   abGeneralData[1];  /*  General data */
 } DRIVDATA;
```

ENTRYFDATA

Entry field control data structure.

```
typedef struct _ENTRYFDATA {
  USHORT   cb;              /*  Length of control data in bytes
                               */
  USHORT   cchEditLimit;    /*  Edit limit */
  USHORT   ichMinSel;       /*  Minimum selection */
  USHORT   ichMaxSel;       /*  Maximum selection */
 } ENTRYFDATA;
```

ERRINFO

Error-information structure.

```
typedef struct _ERRINFO {
  USHORT   cbFixedErrInfo;  /*  Length of fixed data to this
                               structure */
  ERRORID  idError;         /*  Error identity */
  USHORT   cDetailLevel;    /*  Number of levels of detail */
  USHORT   offaoffszMsg;    /*  Offset to the array of message
                               offsets */
  USHORT   offBinaryData;   /*  Offset to the binary data */
 } ERRINFO;
```

ERRORID

Error identity.

```
typedef ULONG ERRORID;
```

FATTRS

Font-attributes structure.

```
typedef struct _FATTRS {
  USHORT   usRecordLength;          /* Length of record */
  USHORT   fsSelection;             /* Selection indicators */
  LONG     lMatch;                  /* Matched-font identity */
  CHAR     szFacename[FACESIZE];    /* Typeface name */
  USHORT   idRegistry;              /* Registry identifier */
  USHORT   usCodePage;              /* Code page */
  LONG     lMaxBaselineExt;         /* Maximum base-line
                                       extension */
  LONG     lAveCharWidth;           /* Average character width
                                       */
  USHORT   fsType;                  /* Type indicators */
  USHORT   fsFontUse;               /* Font-use indicators */
 } FATTRS;
```

FFDESCS

Font-file descriptor.

```
typedef CHAR FFDESCS[2][FACESIZE];
```

FIXED

Signed-integer fraction (16:16). This can be treated as a *LONG* where the value has been multiplied by 65 536.

```
typedef LONG FIXED;
```

FONTMETRICS

Font-metrics structure.

```
typedef struct _FONTMETRICS {
  CHAR     szFamilyname[FACESIZE];     /* Family name */
  CHAR     szFacename[FACESIZE];       /* Facename */
  USHORT   idRegistry;                 /* Registry identifier */
  USHORT   usCodePage;                 /* Code page */
  LONG     lEmHeight;                  /* Em height */
  LONG     lXHeight;                   /* x height */
  LONG     lMaxAscender;               /* Maximum ascender */
  LONG     lMaxDescender;              /* Maximum descender */
  LONG     lLowerCaseAscent;           /* Lowercase ascent */
  LONG     lLowerCaseDescent;          /* Lowercase descent */
  LONG     lInternalLeading;           /* Internal leading */
  LONG     lExternalLeading;           /* External leading */
  LONG     lAveCharWidth;              /* Average character width
                                          */
  LONG     lMaxCharInc;                /* Maximum character
                                          increment */
  LONG     lEmInc;                     /* Em increment */
  LONG     lMaxBaselineExt;            /* Maximum baseline extent
                                          */
  SHORT    sCharSlope;                 /* Character slope */
  SHORT    sInlineDir;                 /* Inline direction */
  SHORT    sCharRot;                   /* Character rotation */
  USHORT   usWeightClass;              /* Weight class */
  USHORT   usWidthClass;               /* Width class */
  SHORT    sXDeviceRes;                /* x device resolution */
  SHORT    sYDeviceRes;                /* y device resolution */
  SHORT    sFirstChar;                 /* First character */
  SHORT    sLastChar;                  /* Last character */
  SHORT    sDefaultChar;               /* Default character */
  SHORT    sBreakChar;                 /* Break character */
  SHORT    sNominalPointSize;          /* Nominal point size */
  SHORT    sMinimumPointSize;          /* Minimum point size */
  SHORT    sMaximumPointSize;          /* Maximum point size */
  USHORT   fsType;                     /* Type indicators */
  USHORT   fsDefn;                     /* Definition indicators */
  USHORT   fsSelection;                /* Selection indicators */
  USHORT   fsCapabilities;             /* Capabilities */
  LONG     lSubscriptXSize;            /* Subscript x size */
  LONG     lSubscriptYSize;            /* Subscript y size */
  LONG     lSubscriptXOffset;          /* Subscript x offset */
  LONG     lSubscriptYOffset;          /* Subscript y offset */
  LONG     lSuperscriptXSize;          /* Superscript x size */
  LONG     lSuperscriptYSize;          /* Superscript y size */
  LONG     lSuperscriptXOffset;        /* Superscript x offset */
  LONG     lSuperscriptYOffset;        /* Superscript y offset */
  LONG     lUnderscoreSize;            /* Underscore size */
  LONG     lUnderscorePosition;        /* Underscore position */
  LONG     lStrikeoutSize;             /* Strikeout size */
  LONG     lStrikeoutPosition;         /* Strikeout position */
  SHORT    sKerningPairs;              /* Kerning pairs */
  SHORT    sFamilyClass;               /* Family class */
  LONG     lMatch;                     /* Matched font identity */
} FONTMETRICS;
```

FRAMECDATA Frame-control data structure.

```
typedef struct _FRAMECDATA {
  USHORT    cb;             /*   Length */
  ULONG     flCreateFlags;  /*   Frame-creation flags */
  HMODULE   hmodResources;  /*   Identifier of required resource
                                */
  USHORT    idResources;    /*   Resource identifier */
 } FRAMECDATA;
```

GRADIENT Direction-vector structure.
(Maps via GRADIENTL)

GRADIENTL Direction-vector structure.

```
typedef struct _GRADIENTL {
  LONG   x;   /*  x component of direction */
  LONG   y;   /*  y component of direction */
 } GRADIENTL;
```

HAB Anchor-block handle.

```
typedef LHANDLE HAB;
```

HACCEL Accelerator-table handle.

```
typedef LHANDLE HACCEL;
```

HANDLE Resource handle.
(Maps via LHANDLE)

HAPP Handle of an application started by the WinInstStartApp function.

```
typedef LHANDLE HAPP;
```

HATOMTBL Atom-table handle.

```
typedef LHANDLE HATOMTBL;
```

HBITMAP Bit-map handle.

```
typedef LHANDLE HBITMAP;
```

HCINFO Hardcopy-capabilities structure.

```
typedef struct _HCINFO {
  CHAR   szFormname[32];    /* Form name */
  LONG   cx;                /* Width (left-to-right) in
                               millimeters */
  LONG   cy;                /* Height (top-to-bottom) in
                               millimeters */
  LONG   xLeftClip;         /* Left clip limit in millimeters */
  LONG   yBottomClip;       /* Bottom clip limit in millimeters */
  LONG   xRightClip;        /* Right clip limit in millimeters */
  LONG   yTopClip;          /* Top clip limit in millimeters */
  LONG   xPels;             /* Number of pels between left and
                               right clip limits */
  LONG   yPels;             /* Number of pels between bottom and
                               top clip limits */
  LONG   flAttributes;      /* Attributes of the form identifier
                               */
 } HCINFO;
```

HDC Device-context handle.

```
typedef LHANDLE HDC;
```

HELPINIT

Help manager initialization structure.

```
typedef struct _HELPINIT {
  USHORT        cb;                       /*  Count of bytes of
                                              the initialization
                                              structure */
  ULONG         ulReturnCode;             /*  Value returned by
                                              the Help Manager from
                                              initialization */
  PSZ           pszTutorialName;          /*  Indicates to the
                                              Help Manager that the
                                              application has a
                                              tutorial program */
  PHELPTABLE    phtHelpTable;             /*  Help table */
  HMODULE       hmodHelpTableModule;      /*  Resource file
                                              identity */
  HMODULE       hmodAccelActionBarModule; /*  Handle of the DLL
                                              that contains the
                                              accelerator table and
                                              action bar template
                                              to be used by the
                                              Help Manager */
  USHORT        idAccelTable;             /*  Identity of the
                                              accelerator table */
  USHORT        idActionBar;              /*  Identity of the
                                              action bar template
                                              used by the Help
                                              Manager */
  PSZ           pszHelpWindowTitle;       /*  Window title for
                                              each help window */
  USHORT        usShowPanelId;            /*  Show panel identity
                                              indicator */
  PSZ           pszHelpLibraryName;       /*  Names of the help
                                              panel libraries that
                                              the Help Manager
                                              searches on each help
                                              request */
 } HELPINIT;
```

HELPSUBTABLE

Help subtable structure.

```
typedef struct _HELPSUBTABLE {
  USHORT              usSubitemSize;              /*  Number of
                                                      words in
                                                      each entry
                                                      of the help
                                                      subtable
                                                      */
  HELPSUBTABLEITEM    ahstiHelpSubTableItems[1]; /*  Array of
                                                      help
                                                      subtable
                                                      entries */
 } HELPSUBTABLE;
```

HELPSUBTABLEITEM Help subtable entry structure.

```
typedef struct _HELPSUBTABLEITEM {
  USHORT    idField;                          /* Identity of the
                                              field from which the
                                              user can request help
                                              */
  USHORT    idHelpPanel;                      /* Identity of the
                                              contextual help panel
                                              */
  USHORT    ausOptionalIntegers[subitemsize-2]; /*
                                              Application-related
                                              integers */
} HELPSUBTABLEITEM;
```

HELPTABLE Help table structure.

This is an array of help table entries, each of which has the structure defined below, the last entry of the array having its **extpanel** parameter set to NULL.

```
typedef struct _HELPTABLE {
  USHORT           idAppWindow;       /* Application window
                                         identity */
  PHELPSUBTABLE    phstHelpSubTable;  /* Help subtable for this
                                         application window */
  USHORT           idExtPanel;        /* Identity of the
                                         extended help panel for
                                         the application window
                                         */
} HELPTABLE;
```

HENUM Window-enumeration handle.

```
typedef LHANDLE HENUM;
```

HHEAP Heap handle.

```
typedef LHANDLE HHEAP;
```

HINI Initialization-file handle.

```
typedef LHANDLE HINI;
```

HLIB Library handle.

```
typedef LHANDLE HLIB;
```

HMF Metafile handle.

```
typedef LHANDLE HMF;
```

HMODULE Module handle.

```
typedef USHORT HMODULE;
```

HMQ Message-queue handle.

```
typedef LHANDLE HMQ;
```

HPOINTER Handle of a pointer.

```
typedef LHANDLE HPOINTER;
```

HPROC Processor handle.

```
typedef LHANDLE HPROC;
```

HPROGARRAY Array of program handles.

```
typedef struct _HPROGARRAY {
  HPROGRAM     ahprog[1];   /* Program handle array */
 } HPROGARRAY;
```

HPROGRAM Program handle.

```
typedef LHANDLE HPROGRAM;
```

HPS Presentation-space handle.

```
typedef LHANDLE HPS;
```

HRGN Region handle.

```
typedef LHANDLE HRGN;
```

HSEM Semaphore handle.

```
typedef VOID FAR *HSEM;
```

HSPL Spooler handle.

```
typedef LHANDLE HSPL;
```

HSWITCH Switch-list entry handle.

```
typedef LHANDLE HSWITCH;
```

HVPS VIO presentation-space handle.

```
typedef USHORT HVPS;
```

HWND Window handle.

```
typedef LHANDLE HWND;
```

IDENTITY Identity value. Up to 65 536 different identities are available. (Maps via USHORT)

IDENTITY4 Identity value. Up to 4 294 967 296. different identities are available. (Maps via ULONG)

IMAGEBUNDLE Image-attributes bundle structure.

```
typedef struct _IMAGEBUNDLE {
  LONG     lColor;           /* Image foreground color */
  LONG     lBackColor;       /* Image background color */
  USHORT   usMixMode;        /* Image foreground-mix mode */
  USHORT   usBackMixMode;    /* Image background-mix mode */
 } IMAGEBUNDLE;
```

INDEX2 Index value, in the range 0 through 65 535. (Maps via USHORT)

IPT Insertion point for multi-line entry field.

```
typedef LONG IPT;
```

KERNINGPAIRS Kerning-pair records structure.

```
typedef struct _KERNINGPAIRS {
  SHORT   sFirstChar;        /* First character of pair */
  SHORT   sSecondChar;       /* Second character of pair */
  SHORT   sKerningAmount;    /* Amount of kerning for this pair
                                */
 } KERNINGPAIRS;
```

LENGTH2 Length value, in the range 0 through 65 535. (Maps via USHORT)

LENGTH4 Length value, in the range 0 through 4 294 967 295. (Maps via ULONG)

LHANDLE The handle of a resource.

```
typedef VOID FAR *LHANDLE;
```

LINEBUNDLE Line-attributes bundle structure.

```
typedef struct _LINEBUNDLE {
  LONG     lColor;        /* Line foreground color */
  LONG     lReserved;     /* Reserved */
  USHORT   usMixMode;     /* Line foreground-mix mode */
  USHORT   usReserved;    /* Reserved */
  FIXED    fxWidth;       /* Line width */
  LONG     lGeomWidth;    /* Geometric line width */
  USHORT   usType;        /* Line type */
  USHORT   usEnd;         /* Line end */
  USHORT   usJoin;        /* Line join */
 } LINEBUNDLE;
```

LONG Signed integer in the range -2 147 483 648 through 2 147 483 647.

Note: Where this data type represents a graphic coordinate in world or model space, its value is restricted to -134 217 728 through 134 217 727. A graphic coordinate in device or screen coordinates is restricted to -32 768 through 32 767. Its value may be further restricted by any transforms currently in force, including the positioning of the origin of the window on the screen. In particular, coordinates in world or model space must not generate coordinate values after transformation (that is in device or screen space) outside the range -32 768 through 32 767.

```
#define LONG long
```

MARGSTRUCT Multi-line entry field margin information

```
typedef struct _MARGSTRUCT {
  USHORT    fsFlags;
  USHORT    usMouMsg;
  IPT       iptNear;
 } MARGSTRUCT;
```

MARKERBUNDLE Marker-attributes bundle structure.

```
typedef struct _MARKERBUNDLE {
  LONG     lColor;           /* Marker foreground color */
  LONG     lBackColor;       /* Marker background color */
  USHORT   usMixMode;        /* Marker foreground-mix mode */
  USHORT   usBackMixMode;    /* Marker background-mix mode */
  USHORT   usSet;            /* Marker set */
  USHORT   usSymbol;         /* Marker symbol */
  SIZEF    sizfxCell;        /* Marker cell */
 } MARKERBUNDLE;
```

MATRIX Matrix-elements structure.
(Maps via MATRIXLF)

MATRIXLF

Matrix-elements structure.

```
typedef struct _MATRIXLF {
  FIXED   fxM11;    /*   First element of first row */
  FIXED   fxM12;    /*   Second element of first row */
  LONG    lM13;     /*   Third element of first row */
  FIXED   fxM21;    /*   First element of second row */
  FIXED   fxM22;    /*   Second element of second row */
  LONG    lM23;     /*   Third element of second row */
  LONG    lM31;     /*   First element of third row */
  LONG    lM32;     /*   Second element of third row */
  LONG    lM33;     /*   Third element of third row */
 } MATRIXLF;
```

MENUITEM

Menu item.

```
typedef struct _MENUITEM {
  SHORT   iPosition;    /*    Position */
  USHORT  afStyle;      /*    Style */
  USHORT  afAttribute;  /*    Attribute */
  USHORT  id;           /*    Identity */
  HWND    hwndSubMenu;  /*    Submenu */
  ULONG   hItem;        /*    Item */
 } MENUITEM;
```

MLECTLDATA

Multi-line entry field control data structure.

```
typedef struct _MLECTLDATA {
  USHORT      cbCtlData;      /* Length of control data in bytes
                                */
  USHORT      afIEFormat;     /* Import/export format */
  ULONG       cchText;        /* Text limit */
  IPT         iptAnchor;      /* Selection anchor point */
  IPT         iptCursor;      /* Selection cursor point */
  LONG        cxFormat;       /* Formatting rectangle width in
                                pixels */
  LONG        cyFormat;       /* Formatting rectangle height in
                                pixels */
  ULONG       afFormatFlags;  /* Format flags */
 } MLECTLDATA;
```

MLE_SEARCHDATA

Search structure for multi-line entry field.

```
typedef struct _MLE_SEARCHDATA {
  SHORT   sStructSize;          /*  Size of MLE_SEARCHDATA
                                    structure */
  SHORT   sFindStringLength;    /*  Length of findstring string
                                    */
  PSZ     pszFindString;        /*  String to search for */
  IPT     iptStart;             /*  Point at which to start
                                    search, or point where string
                                    was found */
  IPT     iptStop;              /*  Point at which to stop
                                    search */
  SHORT   sFoundStringLength;   /*  Length of string found at
                                    start */
  SHORT   sChangeStringLength;  /*  Length of replacement string
                                    */
  PSZ     pszChangeString;      /*  Replacement string */
 } MLE_SEARCHDATA;
```

MT

Menu template.
(Maps via PVOID)

MTI

Menu template item.

```
typedef struct _MTI {
  USHORT    afStyle;   /*   Style */
  USHORT    afAttrs;   /*   Attributes */
  USHORT    idItem;    /*   Item identity */
  CHAR      c[2];      /*   Item data */
 } MTI;
```

MPARAM

4-byte message-dependent parameter structure.

Certain elements of information, placed into the parameters of a message, have data types that do not consume the whole of the 4 bytes of this data type. The rules governing these cases are:

BOOL The value is contained in the low word and the high word is zero.

SHORT The value is contained in the low word and its sign is extended into the high word.

USHORT The value is contained in the low word and the high word is zero.

NULL The entire 4 bytes are zero.

```
typedef VOID FAR *MPARAM;
```

MQINFO

Message-queue information structure.

```
typedef struct _MQINFO {
  USHORT    cb;          /* Length of structure */
  PID       pid;         /* Process identity */
  TID       tid;         /* Thread identity */
  USHORT    cmsgs;       /* Message count */
  PVOID     pReserved;   /* Reserved */
 } MQINFO;
```

MRESULT

4-byte message-dependent reply parameter structure.

Certain elements of information, placed into the parameters of a message, have data types that do not consume the whole of the 4 bytes of this data type. The rules governing these cases are:

BOOL The value is contained in the low word and the high word is zero.

SHORT The value is contained in the low word and its sign is extended into the high word.

USHORT The value is contained in the low word and the high word is zero.

NULL The entire 4 bytes are zero.

```
typedef VOID FAR *MRESULT;
```

NPBYTE

Near pointer to a byte string.

```
typedef BYTE near *NPBYTE;
```

OFFSET2B

Offset value in bytes, in the range 0 through 65 535.
(Maps via USHORT)

OVERFLOW

Overflow error structure for multi-line entry field.

```
typedef struct _OVERFLOW {
  ULONG     ulErrInd;       /*  One or more EFR_* flags */
  LONG      lBytesOver;     /*  Number of bytes over the limit */
  PIX       pixHorzOver;    /*  Number of pixels over the horizontal
                               limit */
  PIX       pixVertOver;    /*  Number of pixels over the vertical
                               limit */
 } OVERFLOW;
```

OWNERITEM Owner item.

```
typedef struct _OWNERITEM {
  HWND     hwnd;          /*   Window handle */
  HPS      hps;           /*   Presentation-space handle */
  USHORT   fsState;       /*   State */
  USHORT   fsAttribute;   /*   Attribute */
  USHORT   fsStateOld;    /*   Old state */
  USHORT   fsAttributeOld; /*  Old attribute */
  RECTL    rclItem;       /*   Item rectangle */
  SHORT    idItem;        /*   Item identity */
  ULONG    hItem;         /*   Item */
 } OWNERITEM;
```

PARAM Presentation parameter attribute definition.

```
typedef struct _PARAM {
  ULONG   id;     /*  Attribute type identity */
  ULONG   cb;     /*  Byte count of the attrvalue parameter */
  BYTE    ab[1]; /*  Attribute value */
 } PARAM;
```

PBUNDLE Points to a bundle data area.

```
typedef PVOID PBUNDLE;
```

PBYTE Pointer to a data area.

```
typedef BYTE FAR *PBYTE;
```

PCH Pointer to a character string.

```
typedef char far *PCH;
```

PDEVOPENDATA Open device-data array.

This data type points to data whose format is described by the *DEVOPENSTRUC* data type.

```
typedef PSZ FAR *PDEVOPENDATA;
```

PFFDESCS Pointer to a font file descriptor.

```
typedef FFDESCS FAR *PFFDESCS;
```

PFN Pointer to procedure.

```
typedef int (pascal far *PFN)();
```

PFNWP Pointer to a window procedure.

```
typedef MRESULT (PASCAL FAR *PFNWP)(HWND, USHORT, MPARAM, MPARAM);
```

PIBSTRUCT Program-information-block structure.

```
typedef struct _PIBSTRUCT {
  PROGTYPE    progt;                          /* Program type and
                                                 visibility */
  CHAR        szTitle[MAXNAMEL+1];            /* Program title
                                                 (null terminated)
                                                 */
  CHAR        szIconFileName[MAXPATHL+1];     /* Program icon
                                                 filename (null
                                                 terminated) */
  CHAR        szExecutable[MAXPATHL+1];       /* Executable file
                                                 name (null
                                                 terminated) */
  CHAR        szStartupDir[MAXPATHL+1];       /* Start-up
                                                 directory (null
                                                 terminated) */
```

```
  XYWINSIZE   xywinInitial;              /* Initial window
                                            position and size
                                            */
  USHORT      res1;                      /* Reserved; must
                                            be zero */
  LHANDLE     res2;                      /* Reserved; must
                                            be zero */
  USHORT      cchEnvironmentVars;        /* Environment
                                            string length */
  PCH         pchEnvironmentVars;        /* Environment
                                            string */
  USHORT      cchProgramParameter;       /* Parameter string
                                            length */
  PCH         pchProgramParameter;       /* Parameter string
                                            */
 } PIBSTRUCT;
```

PID

Process identity.

```
typedef USHORT PID;
```

PIX

Pel count for multi-line entry field.

```
typedef LONG PIX;
```

PMPARAM

Pointer to a 4-byte message-dependent parameter structure.

```
typedef MPARAM FAR *PMPARAM;
```

PMRESULT

Pointer to a 4-byte message-dependent reply parameter structure.

```
typedef MRESULT FAR *PMRESULT;
```

POINT

Point structure.
(Maps via POINTL)

POINTERINFO

Pointer-information structure.

```
typedef struct _POINTERINFO {
  BOOL      fPointer;      /* Bit-map size indicator */
  SHORT     xHotspot;      /* x coordinate of action point */
  SHORT     yHotspot;      /* y coordinate of action point */
  HBITMAP   hbmPointer;    /* Bit-map handle of pointer */
  HBITMAP   hbmColor;      /* Bit-map handle of color bit map */
 } POINTERINFO;
```

POINTL

Point structure (long integer).

```
typedef struct _POINTL {
  LONG   x;   /*  x coordinate */
  LONG   y;   /*  y coordinate */
 } POINTL;
```

POINTS

Point structure (short integer).

```
typedef struct _POINTS {
  SHORT   x;   /*  x coordinate */
  SHORT   y;   /*  y coordinate */
 } POINTS;
```

PQMOPENDATA

Open queue-manager data array.

This data type points to data whose format is described by the *DEVOPENSTRUC* data type.

```
typedef PSZ FAR *PQMOPENDATA;
```

PRESPARAMS

Presentation parameter data.

```
typedef struct _PRESPARAMS {
  ULONG    cb;          /* Byte count of the param parameter */
  PARAM    aparam[1];   /* Array of attribute parameters */
} PRESPARAMS;
```

PRESDATA

Class-specific control presentation data, conforming to a *PRESPARAM* data type.
(Maps via PVOID)

PRFPROFILE

Profile structure.

```
typedef struct _PRFPROFILE {
  ULONG    cchUserName;     /* Length of user profile name */
  PSZ      pszUserName;     /* User profile name */
  ULONG    cchSysName;      /* Length of system profile name */
  PSZ      pszSysName;      /* System profile name */
} PRFPROFILE;
```

PROC

Procedure identifier.
(Maps via PFN)

PROGCATEGORY

Program category.

```
typedef CHAR PROGCATEGORY;
```

PROGDETAILS

Program details structure.

```
typedef struct _PROGDETAILS {
  ULONG       Length;          /* Length of structure */
  PROGTYPE    progt;           /* Program type */
  USHORT      pad1[3];         /* Reserved */
  PSZ         pszTitle;        /* Title */
  PSZ         pszExecutable;   /* Executable file name */
  PSZ         pszParameters;   /* Parameter string */
  PSZ         pszStartupDir;   /* Start-up directory */
  PSZ         pszIcon;         /* Icon-file name */
  PSZ         pszEnvironment;  /* Environment string. (List of
                                  null-terminated strings, ending
                                  with an extra null) */
  SWP         swpInitial;      /* Initial window position and
                                  size */
  USHORT      pad2[5];         /* Reserved */
} PROGDETAILS;
```

PROGRAMENTRY

Program-entry structure.

```
typedef struct _PROGRAMENTRY {
  HPROGRAM     hprog;                  /* Program handle */
  PROGTYPE     progt;                  /* Program type */
  CHAR         szTitle[MAXNAMEL+1];    /* Program title (null
                                          terminated) */
} PROGRAMENTRY;
```

PROGTITLE

Program-title structure.

```
typedef struct _PROGTITLE {
  HPROGRAM     hprog;     /* Program handle */
  PROGTYPE     progt;     /* Program type */
  USHORT       pad1[3];   /* Reserved */
  PSZ          pszTitle;  /* Program title */
} PROGTITLE;
```

PROGTYPE

Program-type structure.

```
typedef struct _PROGTYPE {
  PROGCATEGORY      progc;      /* Program category */
  UCHAR             fbVisible;  /* Visibility attribute */
 } PROGTYPE;
```

PROPERTY2

Property value. Up to 65 536 different properties are available.
(Maps via USHORT)

PROPERTY4

Property value. Up to 4 294 967 296 different properties are available.
(Maps via LONG)

PSZ

Pointer to a null-terminated string.

```
typedef char far *PSZ;
```

PVOID

Pointer to a data type of undefined format.

```
typedef VOID FAR *PVOID;
```

QMOPENDATA

Open queue-manager data area. The format of this area is the same as a *DEVOPENSTRUC* structure.
(Maps via PQMOPENDATA)

QMSG

Message structure.

```
typedef struct _QMSG {
  HWND     hwnd;     /* Window handle */
  USHORT   msg;      /* Message identity */
  MPARAM   mp1;      /* Parameter 1 */
  MPARAM   mp2;      /* Parameter 2 */
  ULONG    time;     /* Message time */
  POINTL   ptl;      /* Pointer position when message was generated
                        */
 } QMSG;
```

RECT

Rectangle structure.

Unless otherwise stated, points on the right-hand or top boundaries of the rectangle are not considered to be included in the rectangle; points on the left-hand or bottom boundaries (that are not also on the right-hand or top boundaries) are included in the rectangle.
(Maps via RECTL)

RECTL

Rectangle structure.

```
typedef struct _RECTL {
  LONG   xLeft;     /* x coordinate of left-hand edge of rectangle
                       */
  LONG   yBottom;   /* y coordinate of bottom edge of rectangle
                       */
  LONG   xRight;    /* x coordinate of right-hand edge of
                       rectangle */
  LONG   yTop;      /* y coordinate of top edge of rectangle */
 } RECTL;
```

RESID

Resource identity.
(Maps via HMODULE)

RGB

RGB color value.

```
typedef struct _RGB {
  BYTE   bBlue;    /*  Blue component of the color definition */
  BYTE   bGreen;   /*  Green component of the color definition */
  BYTE   bRed;     /*  Red component of the color definition */
 } RGB;
```

RGNRECT

Region-rectangle structure.

```
typedef struct _RGNRECT {
  USHORT    ircStart;      /*  Rectangle number from which to start
                               enumerating */
  USHORT    crc;           /*  Number of rectangles that can be
                               returned */
  USHORT    crcReturned;   /*  Number of rectangles returned */
  USHORT    usDirection;   /*  Direction in which the returned
                               rectangles are to be ordered */
} RGNRECT;
```

ROF

Number representation, equivalent to *FIXED*.
(Maps via FIXED)

ROL

Number representation, equivalent to *LONG*.
(Maps via LONG)

SBCDATA

Scroll-bar control data structure.

```
typedef struct _SBCDATA {
  USHORT    cb;         /*  Length of control data in bytes */
  USHORT    sHilite;    /*  Highlighting code */
  SHORT     posFirst;   /*  First bound of the scroll-bar range */
  SHORT     posLast;    /*  Last bound of the scroll-bar range */
  SHORT     posThumb;   /*  Slider position */
  SHORT     cVisible;   /*  Number of data items visible */
  SHORT     cTotal;     /*  Number of data items available */
} SBCDATA;
```

SEGOFF

2-byte segment offset in bytes.
(Maps via NPBYTE)

SFACTORS

Scaling factors, see DevEscape.

```
typedef struct _SFACTORS {
  LONG   lx;   /*  x scaling factor, as an exponent of 2 */
  LONG   ly;   /*  y scaling factor, as an exponent of 2 */
} SFACTORS;
```

SHORT

Signed integer in the range -32 768 through 32 767.

```
#define SHORT short
```

SIZEF

Size structure.

```
typedef struct _SIZEF {
  FIXED  cx;   /*  Width */
  FIXED  cy;   /*  Height */
} SIZEF;
```

SIZEL

Size structure.

```
typedef struct _SIZEL {
  LONG   cx;   /*  Width */
  LONG   cy;   /*  Height */
} SIZEL;
```

SIZEROF

Size structure.
(Maps via SIZEF)

SIZEROL

Size structure.
(Maps via SIZEL)

SMHSTRUCT Send message hook structure.

```
typedef struct _SMHSTRUCT {
  MPARAM     mp2;     /* Parameter 2 */
  MPARAM     mp1;     /* Parameter 1 */
  USHORT     msg;     /* Message identity */
  HWND       hwnd;    /* Window handle */
 } SMHSTRUCT;
```

STORAGE Storage that may contain different data types.
(Maps via PVOID)

STR String with an explicit length count.
(Maps via PCH)

STRCOND String with a length count that is determined by certain values in its associated length parameter.

For example, positive values in the associated length parameter provide the length count explicitly, whereas the value **–1** implies that the string is null-terminated and hence its length is implicit.

Where this data type occurs, the associated length parameter specifies the meanings of its values in determining the length count of the string.
(Maps via PCH)

STRL Null-terminated string, that is, its length can be implied by the system.

(Maps via SZ)

STRLLIST A list of null-terminated strings (that is, a series of concatenated strings each of which has a *STRL* data type) terminated by an additional null.
(Maps via SZ)

STR8 String of 8 characters.

```
typedef CHAR STR8[8];
```

STR16 String of characters, with an implicit length, in a 16-byte field.

```
typedef CHAR STR16[16];
```

STR32 String of characters, with an implicit length, in a 32-byte field.

```
typedef CHAR STR32[32];
```

STR64 String of characters, with an implicit length, in a 64-byte field.

```
typedef CHAR STR64[64];
```

SWBLOCK Switch list block structure.

```
typedef struct _SWBLOCK {
  USHORT    cswentry;     /*   Count of switch list entries */
  SWENTRY   aswentry[1];  /*   Switch list entries */
 } SWBLOCK;
```

SWCNTRL Switch-list control block structure.

```
typedef struct _SWCNTRL {
  HWND          hwnd;                    /* Window handle */
  HWND          hwndIcon;                /* Window-handle icon */
  HPROGRAM      hprog;                   /* Program handle */
  USHORT        idProcess;               /* Process identity */
  USHORT        idSession;               /* Session identity */
  UCHAR         uchVisibility;           /* Visibility */
  UCHAR         fbJump;                  /* Jump indicator */
  CHAR          szSwtitle[MAXNAMEL+1];   /* Switch-list control
                                            block title (null
                                            terminated) */
  BYTE          fReserved;               /* Reserved */
 } SWCNTRL;
```

SWENTRY

Switch list entry structure.

```
typedef struct _SWENTRY {
  HSWITCH    hswitch;    /*  Switch list entry handle */
  SWCNTRL    swctl;      /*  Switch list control block structure */
 } SWENTRY;
```

SWP

Set Window Position structure.

```
typedef struct _SWP {
  USHORT     fs;                 /*  Options */
  SHORT      cy;                 /*  Window height */
  SHORT      cx;                 /*  Window width */
  SHORT      y;                  /*  y coordinate of origin */
  SHORT      x;                  /*  x coordinate of origin */
  HWND       hwndInsertBehind;   /*  Window behind which this
                                     window is placed */
  HWND       hwnd;               /*  Window handle */
 } SWP;
```

SZ

Null-terminated string (also known as a zero-terminated string). (Maps via PSZ)

TID

Thread identity.

```
typedef USHORT TID;
```

TIME

Time interval in milliseconds. (Maps via LONG)

TRACKINFO

Tracking-information structure.

```
typedef struct _TRACKINFO {
  SHORT    cxBorder;          /*   Border width */
  SHORT    cyBorder;          /*   Border height */
  SHORT    cxGrid;            /*   Grid width */
  SHORT    cyGrid;            /*   Grid height */
  SHORT    cxKeyboard;        /*   Character cell width movement
                                  for arrow key */
  SHORT    cyKeyboard;        /*   Character cell height movement
                                  for arrow key */
  RECTL    rclTrack;          /*   Starting tracking rectangle */
  RECTL    rclBoundary;       /*   Boundary rectangle */
  POINTL   ptlMinTrackSize;   /*   Minimum tracking size */
  POINTL   ptlMaxTrackSize;   /*   Maximum tracking size */
  USHORT   fs;                /*   Tracking options */
 } TRACKINFO;
```

UCHAR

Unsigned integer in the range 0 through 255.

```
typedef unsigned char UCHAR;
```

ULONG

Unsigned integer in the range 0 through 4 294 967 295.

```
typedef unsigned long ULONG;
```

USERBUTTON

User-button data structure.

```
typedef struct _USERBUTTON {
  HWND      hwnd;        /*  Window handle */
  HPS       hps;         /*  Presentation-space handle */
  USHORT    fsState;     /*  New state of user button */
  USHORT    fsStateOld;  /*  Old state of user button */
 } USERBUTTON;
```

USHORT

Unsigned integer in the range 0 through 65 535.

```
typedef unsigned short USHORT;
```

VIOSIZECOUNT

Count of VIO cell sizes, see DevEscape.

```
typedef struct _VIOSIZECOUNT {
  LONG   maxcount;     /* Maximum number of VIO cell sizes
                          supported */
  LONG   count;        /* Number of VIO cell sizes returned */
 } VIOSIZECOUNT;
```

VIOFONTCELLSIZE

VIO cell size, see DevEscape.

```
typedef struct _VIOFONTCELLSIZE {
  LONG   cx;   /* Cell width */
  LONG   cy;   /* Cell height */
 } VIOFONTCELLSIZE;
```

VOID

A data area of undefined format.

```
#define VOID void
```

WIDTH4

Width in the range -2 147 483 648 through 2 147 483 647.
(Maps via LONG)

WNDPARAMS

Window parameters.

```
typedef struct _WNDPARAMS {
  USHORT    fsStatus;        /* Window parameter selection */
  USHORT    cchText;         /* Length of window text */
  PSZ       pszText;         /* Window text */
  USHORT    cbPresParams;    /* Length of presentation parameters
                                */
  PVOID     pPresParams;     /* Presentation parameters */
  USHORT    cbCtlData;       /* Length of window class specific
                                data */
  PVOID     pCtlData;        /* Window class specific data */
 } WNDPARAMS;
```

WNDPROC

Window-procedure identifier.
(Maps via PFNWP)

WPOINT

Window point structure (integer).

```
typedef struct _WPOINT {
  SHORT   x;          /* x coordinate */
  SHORT   dummy1;     /* Reserved */
  SHORT   y;          /* y coordinate */
  SHORT   dummy2;     /* Reserved */
 } WPOINT;
```

WRECT

Window rectangle structure (integer).

```
typedef struct _WRECT {
  SHORT   xLeft;      /* x coordinate of left-hand edge of
                         rectangle */
  SHORT   dummy1;     /* Reserved */
  SHORT   yBottom;    /* y coordinate of bottom edge of rectangle
                         */
  SHORT   dummy2;     /* Reserved */
  SHORT   xRight;     /* x coordinate of right-hand edge of
                         rectangle */
  SHORT   dummy3;     /* Reserved */
  SHORT   yTop;       /* y coordinate of top edge of rectangle */
  SHORT   dummy4;     /* Reserved */
 } WRECT;
```

XYWINSIZE

Window position and size structure.

```
typedef struct _XYWINSIZE {
  SHORT   x;          /* x coordinate of window origin */
  SHORT   y;          /* y coordinate of window origin */
  SHORT   cx;         /* Window width */
  SHORT   cy;         /* Window height */
  USHORT  fsWindow;   /* Window options */
 } XYWINSIZE;
```

Chapter 3. Device Function Calls

DevCloseDC

```
#define INCL_DEV   /* Or use INCL_PM. Also in COMMON section */

HMF   hmf = DevCloseDC (hdc)

HDC   hdc;  /* Device-context handle */

HMF   hmf;  /* Error indicator/metafile handle (for a metafile device context) */
```

DevEscape

```
#define INCL_DEV   /* Or use INCL_PM */

LONG   lResult = DevEscape (hdc, lCode, lInCount, pbInData, plOutCount, pbOutData)

HDC    hdc;        /* Device-context handle */
LONG   lCode;      /* Escape code */
LONG   lInCount;   /* Input data count */
PBYTE  pbInData;   /* The input data required for this escape */
PLONG  plOutCount; /* Output data count */
PBYTE  pbOutData;  /* Output data */

LONG   lResult;    /* Implemented/error indicator */
```

DevOpenDC

```
#define INCL_DEV   /* Or use INCL_PM. Also in COMMON section */

HDC           hdc = DevOpenDC (hab, lType, pszToken, lCount, pdopData, hdcComp)

HAB           hab;      /* Anchor-block handle */
LONG          lType;    /* Type of device context */
PSZ           pszToken; /* Device-information token */
LONG          lCount;   /* Number of items */
PDEVOPENDATA  pdopData; /* Open-device-context data area */
HDC           hdcComp;  /* Compatible-device-context handle */

HDC           hdc;      /* Device-context handle */
```

DevPostDeviceModes

```
#define INCL_DEV   /* Or use INCL_PM */

LONG       lDriverCount = DevPostDeviceModes (hab, pdrivDriverData, pszDriverName,
                                              pszDeviceName, pszName, flOptions)

HAB        hab;            /* Anchor-block handle */
PDRIVDATA  pdrivDriverData; /* Driver data */
PSZ        pszDriverName;   /* This is a string containing the name of the presentation
                               device driver */
PSZ        pszDeviceName;   /* Device-type name */
PSZ        pszName;         /* Device name */
ULONG      flOptions;       /* Dialog options */

LONG       lDriverCount;    /* Size/error indicator */
```

DevQueryCaps

```
#define INCL_DEV   /* Or use INCL_PM. Also in COMMON section */

BOOL   fSuccess = DevQueryCaps (hdc, lStart, lCount, alArray)

HDC    hdc;      /* Device-context handle */
LONG   lStart;   /* First item of information */
LONG   lCount;   /* Count of items of information */
PLONG  alArray;  /* Device capabilities */

BOOL   fSuccess; /* Success indicator */
```

DevQueryDeviceNames

```
#define INCL_DEV   /* Or use INCL_PM */

BOOL    fSuccess = DevQueryDeviceNames (hab, pszDriverName, pldn, aDeviceName, aDeviceDesc,
                                        pldt, aDataType)

HAB     hab;           /* Anchor-block handle */
PSZ     pszDriverName; /* Fully qualified name of the file containing the presentation
                          driver */
PLONG   pldn;          /* Maximum number of device names and descriptions that can be
                          returned */
PSTR32  aDeviceName;   /* Device-name array */
PSTR64  aDeviceDesc;   /* Device description array */
PLONG   pldt;          /* Maximum number of data types that can be returned */
PSTR16  aDataType;     /* Data type array */

BOOL    fSuccess;      /* Success indicator */
```

DevQueryHardcopyCaps

```
#define INCL_DEV   /* Or use INCL_PM */

LONG     lFormsReturned = DevQueryHardcopyCaps (hdc, lStartForm, lForms, phciHcInfo)

HDC      hdc;            /* Device-context handle */
LONG     lStartForm;     /* Start-forms code */
LONG     lForms;         /* Number of forms to query */
PHCINFO  phciHcInfo;     /* Hardcopy capabilities information */

LONG     lFormsReturned; /* Details of forms */
```

Chapter 4. Graphics Presentation Interface Function Calls

GpiAssociate

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL  fSuccess = GpiAssociate (hps, hdc)

HPS   hps;      /* Presentation-space handle */
HDC   hdc;      /* Device-context handle */

BOOL  fSuccess; /* Success indicator */
```

GpiBeginArea

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL   fSuccess = GpiBeginArea (hps, flOptions)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Area options */

BOOL   fSuccess;  /* Success indicator */
```

GpiBeginElement

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiBeginElement (hps, lType, pszDesc)

HPS   hps;      /* Presentation-space handle */
LONG  lType;    /* Type to be associated with element */
PSZ   pszDesc;  /* Description */

BOOL  fSuccess; /* Success indicator */
```

GpiBeginPath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiBeginPath (hps, lPath)

HPS   hps;      /* Presentation-space handle */
LONG  lPath;    /* Path identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiBitBlt

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG     lHits = GpiBitBlt (hpsTarget, hpsSource, lCount, aptlPoints, lRop, flOptions)

HPS      hpsTarget;  /* Target presentation-space handle */
HPS      hpsSource;  /* Source presentation-space handle */
LONG     lCount;     /* Point count */
PPOINTL  aptlPoints; /* Point array */
LONG     lRop;       /* Mixing function required */
ULONG    flOptions;  /* Options */

LONG     lHits;      /* Correlation/error indicator */
```

GpiBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG     lHits = GpiBox (hps, lControl, pptlPoint, lHRound, lVRound)

HPS      hps;       /* Presentation-space handle */
LONG     lControl;  /* Outline and fill control */
PPOINTL  pptlPoint; /* Corner point */
LONG     lHRound;   /* Corner-rounding control */
LONG     lVRound;   /* Corner-rounding control */

LONG     lHits;     /* Correlation/error indicator */
```

GpiCallSegmentMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

LONG       lHits = GpiCallSegmentMatrix (hps, lSegment, lCount, pmatlfArray, lOptions)

HPS        hps;         /* Presentation-space handle */
LONG       lSegment;    /* Identifier of segment to be called */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfArray; /* Instance transform matrix */
LONG       lOptions;    /* Transformation options */

LONG       lHits;       /* Correlation/error indicator */
```

GpiCharString

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG  lHits = GpiCharString (hps, lCount, pchString)

HPS   hps;       /* Presentation-space handle */
LONG  lCount;    /* Number of characters */
PCH   pchString; /* Characters to be drawn */

LONG  lHits;     /* Correlation/error indicator */
```

GpiCharStringAt

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG     lHits = GpiCharStringAt (hps, pptlPoint, lCount, pchString)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Starting position */
LONG     lCount;    /* Number of characters */
PCH      pchString; /* Characters to be drawn */

LONG     lHits;     /* Correlation/error indicator */
```

GpiCharStringPos

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG    lHits = GpiCharStringPos (hps, prclRect, flOptions, lCount, pchString, alAdx)

HPS     hps;       /* Presentation-space handle */
PRECTL  prclRect;  /* Rectangle structure */
ULONG   flOptions; /* Formatting options */
LONG    lCount;    /* Number of characters in the string */
PCH     pchString; /* Character string */
PLONG   alAdx;     /* Increment values */

LONG    lHits;     /* Correlation/error indicator */
```

GpiCharStringPosAt

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiCharStringPosAt (hps, pptlStart, prclRect, flOptions, lCount, pchString,
                                     alAdx)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlStart; /* Starting position */
PRECTL   prclRect;  /* Rectangle structure */
ULONG    flOptions; /* Formatting options */
LONG     lCount;    /* Number of characters in the string */
PCH      pchString; /* Character string */
PLONG    alAdx;     /* Increment values */

LONG     lHits;     /* Correlation/error indicator */
```

GpiCloseFigure

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiCloseFigure (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiCloseSegment

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiCloseSegment (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiCombineRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG  lComplexity = GpiCombineRegion (hps, hrgnDest, hrgnSrc1, hrgnSrc2, lMode)

HPS   hps;         /* Presentation-space handle */
HRGN  hrgnDest;    /* Destination-region handle */
HRGN  hrgnSrc1;    /* First-source-region handle */
HRGN  hrgnSrc2;    /* Second-source-region handle */
LONG  lMode;       /* Method of combination */

LONG  lComplexity; /* Complexity of resulting region/error indicator */
```

GpiComment

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiComment (hps, lLength, pbData)

HPS    hps;      /* Presentation-space handle */
LONG   lLength;  /* Data length */
PBYTE  pbData;   /* Comment data */

BOOL   fSuccess; /* Success indicator */
```

GpiConvert

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiConvert (hps, lSrc, lTarg, lCount, aptlPoints)

HPS      hps;        /* Presentation-space handle */
LONG     lSrc;       /* Source-coordinate space */
LONG     lTarg;      /* Target-coordinate space */
LONG     lCount;     /* Point count */
PPOINTL  aptlPoints; /* Array of (x,y) coordinate pair structures */

BOOL     fSuccess;   /* Success indicator */
```

GpiCopyMetaFile

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

HMF   hmfNew = GpiCopyMetaFile (hmf)

HMF   hmf;    /* Source metafile handle */

HMF   hmfNew; /* New metafile handle/error indicator */
```

GpiCorrelateChain

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

LONG     lNumHits = GpiCorrelateChain (hps, lType, pptlPick, lMaxHits, lMaxDepth, alSegTag)

HPS      hps;       /* Presentation-space handle */
LONG     lType;     /* Segment type */
PPOINTL  pptlPick;  /* Pick position */
LONG     lMaxHits;  /* Maximum hits */
LONG     lMaxDepth; /* Number of pairs */
PLONG    alSegTag;  /* Segment identifiers and tags */

LONG     lNumHits;  /* Number of hits or error */
```

GpiCorrelateFrom

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

LONG     lNumHits = GpiCorrelateFrom (hps, lFirstSegment, lLastSegment, lType, pptlPick,
                                      lMaxHits, lMaxDepth, alSegTag)

HPS      hps;           /* Presentation-space handle */
LONG     lFirstSegment; /* Specifies the first segment to be correlated */
LONG     lLastSegment;  /* Specifies the last segment to be correlated */
LONG     lType;         /* The type of segments on which correlation is to be performed */
PPOINTL  pptlPick;      /* Pick position */
LONG     lMaxHits;      /* Maximum hits */
LONG     lMaxDepth;     /* Number of pairs */
PLONG    alSegTag;      /* An array of segment and tag identifiers in alternate elements */

LONG     lNumHits;      /* Number of hits, or error */
```

GpiCorrelateSegment

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

LONG     lNumHits = GpiCorrelateSegment (hps, lSegment, lType, pptlPick, lMaxHits, lMaxDepth,
                                         alSegTag)

HPS      hps;       /* Presentation-space handle */
LONG     lSegment;  /* Identifier of the segment to be correlated; it must be greater than 0
                       */
LONG     lType;     /* Type of segments on which correlation is to be performed */
PPOINTL  pptlPick;  /* Position of the center of the pick aperture in presentation page
                       space */
LONG     lMaxHits;  /* Maximum hits */
LONG     lMaxDepth; /* Number of segment and tag pairs to be returned for each hit */
PLONG    alSegTag;  /* Array */

LONG     lNumHits;  /* Number of hits, or error */
```

GpiCreateBitmap

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

HBITMAP            hbm = GpiCreateBitmap (hps, pbmpNew, flOptions, pbInitData, pbmiInfoTable)

HPS                hps;           /* Presentation-space handle */
PBITMAPINFOHEADER  pbmpNew;       /* Bit-map information header */
ULONG              flOptions;     /* Options */
PBYTE              pbInitData;    /* Buffer address */
PBITMAPINFO        pbmiInfoTable; /* Bit-map information table */

HBITMAP            hbm;           /* Bit-map handle/error indicator */
```

GpiCreateLogColorTable

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiCreateLogColorTable (hps, flOptions, lFormat, lStart, lCount, alTable)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Options */
LONG   lFormat;   /* Format of entries in the table */
LONG   lStart;    /* Starting index */
LONG   lCount;    /* Count of elements in Table */
PLONG  alTable;   /* Start of the application data area */

BOOL   fSuccess;  /* Success indicator */
```

GpiCreateLogFont

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

LONG     lMatch = GpiCreateLogFont (hps, pName, lLcid, pfatAttrs)

HPS      hps;       /* Presentation-space handle */
PSTR8    pName;     /* Logical font name */
LONG     lLcid;     /* Local identifier */
PFATTRS  pfatAttrs; /* Attributes of font */

LONG     lMatch;    /* Match indicator */
```

GpiCreatePS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

HPS     hps = GpiCreatePS (hab, hdc, psizlSize, flOptions)

HAB     hab;       /* Anchor-block handle */
HDC     hdc;       /* Device-context handle */
PSIZEL  psizlSize; /* Presentation-page size */
ULONG   flOptions; /* Options */

HPS     hps;       /* Presentation-space handle */
```

GpiCreateRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

HRGN    hrgn = GpiCreateRegion (hps, lCount, arclRectangles)

HPS     hps;            /* Presentation-space handle */
LONG    lCount;         /* The number of rectangles */
PRECTL  arclRectangles; /* An array of rectangles */

HRGN    hrgn;           /* Region handle */
```

GpiDeleteBitmap

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL      fSuccess = GpiDeleteBitmap (hbm)

HBITMAP   hbm;      /* Handle of bit map to be deleted */

BOOL      fSuccess; /* Success indicator */
```

GpiDeleteElement

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteElement (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiDeleteElementRange

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteElementRange (hps, lFirstElement, lLastElement)

HPS   hps;           /* Presentation-space handle */
LONG  lFirstElement; /* Number of the first element to be deleted */
LONG  lLastElement;  /* Number of the last element to be deleted */

BOOL  fSuccess;      /* Success indicator */
```

GpiDeleteElementsBetweenLabels

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteElementsBetweenLabels (hps, lFirstLabel, lLastLabel)

HPS   hps;         /* Presentation-space handle */
LONG  lFirstLabel; /* Label marking the start of the elements to be deleted */
LONG  lLastLabel;  /* Label marking the end of the elements to be deleted */

BOOL  fSuccess;    /* Success indicator */
```

GpiDeleteMetaFile

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteMetaFile (hmf)

HMF   hmf;      /* Metafile handle */

BOOL  fSuccess; /* Success indicator */
```

GpiDeleteSegment

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteSegment (hps, lSegid)

HPS   hps;      /* Presentation-space handle */
LONG  lSegid;   /* Segment identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiDeleteSegments

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteSegments (hps, lFirstSegment, lLastSegment)

HPS   hps;           /* Presentation-space handle */
LONG  lFirstSegment; /* First identifier in the range; it must be greater than 0 */
LONG  lLastSegment;  /* Last identifier in the range; it must be greater than 0 */

BOOL  fSuccess;      /* Success indicator */
```

GpiDeleteSetId

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDeleteSetId (hps, lLcid)

HPS   hps;      /* Presentation-space handle */
LONG  lLcid;    /* Local identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiDestroyPS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL  fSuccess = GpiDestroyPS (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiDestroyRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDestroyRegion (hps, hrgn)

HPS   hps;      /* Presentation-space handle */
HRGN  hrgn;     /* Handle of region to be destroyed */

BOOL  fSuccess; /* Success indicator */
```

GpiDrawChain

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDrawChain (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiDrawDynamics

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDrawDynamics (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiDrawFrom

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDrawFrom (hps, lFirstSegment, lLastSegment)

HPS   hps;           /* Presentation-space handle */
LONG  lFirstSegment; /* First segment to be drawn; it must be greater than zero */
LONG  lLastSegment;  /* Last segment to be drawn; it must be greater than zero */

BOOL  fSuccess;      /* Success indicator */
```

GpiDrawSegment

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiDrawSegment (hps, lSegment)

HPS   hps;      /* Presentation-space handle */
LONG  lSegment; /* Segment to be drawn; it must be greater than zero */

BOOL  fSuccess; /* Success indicator */
```

GpiElement

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

LONG   lHits = GpiElement (hps, lType, pszDesc, lLength, pbData)

HPS    hps;     /* Presentation-space handle */
LONG   lType;   /* Type to be associated with the element */
PSZ    pszDesc; /* Element description */
LONG   lLength; /* Length of content data for the element */
PBYTE  pbData;  /* Buffer pointer */

LONG   lHits;   /* Correlation/error indicator */
```

GpiEndArea

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG  lHits = GpiEndArea (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lHits; /* Correlation/error indicator */
```

GpiEndElement

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiEndElement (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiEndPath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiEndPath (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiEqualRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG  lEquality = GpiEqualRegion (hps, hrgnSrc1, hrgnSrc2)

HPS   hps;       /* Presentation-space handle */
HRGN  hrgnSrc1;  /* Handle of first region */
HRGN  hrgnSrc2;  /* Handle of second region */

LONG  lEquality; /* Equality/error indicator */
```

GpiErase

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL  fSuccess = GpiErase (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiErrorSegmentData

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

LONG   lOff = GpiErrorSegmentData (hps, plSegment, plContext)

HPS    hps;       /* Presentation-space handle */
PLONG  plSegment; /* Segment in which the error occurred */
PLONG  plContext; /* Context of the error */

LONG   lOff;      /* Position */
```

GpiExcludeClipRectangle

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG    lComplexity = GpiExcludeClipRectangle (hps, prclRectangle)

HPS     hps;           /* Presentation-space handle */
PRECTL  prclRectangle; /* Rectangle to be excluded */

LONG    lComplexity;   /* Complexity of clipping/error indicator */
```

GpiFillPath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

LONG  lHits = GpiFillPath (hps, lPath, lOptions)

HPS   hps;      /* Presentation-space handle */
LONG  lPath;    /* Identifier of path whose interior is to be drawn; it must be 1 */
LONG  lOptions; /* Fill option */

LONG  lHits;    /* Error indicator */
```

GpiFullArc

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG   lHits = GpiFullArc (hps, lControl, fxMultiplier)

HPS    hps;          /* Presentation-space handle */
LONG   lControl;     /* Interior/outline control */
FIXED  fxMultiplier; /* Multiplier */

LONG   lHits;        /* Correlation/error indicator */
```

GpiGetData

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

LONG   lCount = GpiGetData (hps, lSegid, plOffset, lFormat, lLength, pbData)

HPS    hps;      /* Presentation-space handle */
LONG   lSegid;   /* Segment identifier */
PLONG  plOffset; /* Segment offset */
LONG   lFormat;  /* Coordinate type required */
LONG   lLength;  /* Length of data buffer */
PBYTE  pbData;   /* Data buffer */

LONG   lCount;   /* Length of returned data */
```

GpiImage

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG    lHits = GpiImage (hps, lFormat, psizlImageSize, lLength, pbData)

HPS     hps;            /* Presentation-space handle */
LONG    lFormat;        /* Format of image data */
PSIZEL  psizlImageSize; /* Size of image area (in pels) */
LONG    lLength;        /* Length in bytes of image data */
PBYTE   pbData;         /* Image data */

LONG    lHits;          /* Correlation/error indicator */
```

GpiIntersectClipRectangle

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG    lComplexity = GpiIntersectClipRectangle (hps, prclRectangle)

HPS     hps;           /* Presentation-space handle */
PRECTL  prclRectangle; /* Rectangle, the coordinates of which are world coordinates */

LONG    lComplexity;   /* Complexity of clipping/error indicator */
```

GpiLabel

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiLabel (hps, lLabel)

HPS   hps;      /* Presentation-space handle */
LONG  lLabel;   /* Required label */

BOOL  fSuccess; /* Success indicator */
```

GpiLine

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG     lHits = GpiLine (hps, pptlEndPoint)

HPS      hps;          /* Presentation-space handle */
PPOINTL  pptlEndPoint; /* End point of the line */

LONG     lHits;        /* Correlation/error indicator */
```

GpiLoadBitmap

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

HBITMAP  hbm = GpiLoadBitmap (hps, Resource, idBitmap, lWidth, lHeight)

HPS      hps;      /* Presentation-space handle */
HMODULE  Resource; /* Resource identity containing the bit map */
USHORT   idBitmap; /* ID of the bit map within the resource file */
LONG     lWidth;   /* Width of the bit map in pels */
LONG     lHeight;  /* Height of the bit map in pels */

HBITMAP  hbm;      /* Bit-map handle */
```

GpiLoadFonts

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiLoadFonts (hab, pszFilename)

HAB   hab;         /* Anchor-block handle */
PSZ   pszFilename; /* Filename */

BOOL  fSuccess;    /* Success indicator */
```

GpiLoadMetaFile

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

HMF   hmf = GpiLoadMetaFile (hab, pszFilename)

HAB   hab;         /* Anchor-block handle */
PSZ   pszFilename; /* Filename */

HMF   hmf;         /* Metafile handle or error */
```

GpiMarker

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiMarker (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Position of the marker */

LONG     lHits;     /* Correlation/error indicator */
```

GpiModifyPath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiModifyPath (hps, lPath, lMode)

HPS   hps;      /* Presentation-space handle */
LONG  lPath;    /* Path identifier */
LONG  lMode;    /* Modification required */

BOOL  fSuccess; /* Success indicator */
```

GpiMove

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL     fSuccess = GpiMove (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Position to which to move */

BOOL     fSuccess;  /* Success indicator */
```

GpiOffsetClipRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG     lComplexity = GpiOffsetClipRegion (hps, pptlPoint)

HPS      hps;         /* Presentation-space handle */
PPOINTL  pptlPoint;   /* Displacement */

LONG     lComplexity; /* Complexity of clipping/error indicator */
```

GpiOffsetElementPointer

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiOffsetElementPointer (hps, loffset)

HPS   hps;      /* Presentation-space handle */
LONG  loffset;  /* Offset to be added to the element pointer */

BOOL  fSuccess; /* Success indicator */
```

GpiOffsetRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiOffsetRegion (hps, Hrgn, pptlOffset)

HPS      hps;        /* Presentation-space handle */
HRGN     Hrgn;       /* Region handle to be moved */
PPOINTL  pptlOffset; /* Offset to be added to the region boundary */

BOOL     fSuccess;   /* Success indicator */
```

GpiOpenSegment

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiOpenSegment (hps, lSegment)

HPS   hps;      /* Presentation-space handle */
LONG  lSegment; /* Segment identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiOutlinePath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

LONG  lHits = GpiOutlinePath (hps, lPath, lOptions)

HPS   hps;      /* Presentation-space handle */
LONG  lPath;    /* Identifier of path to be outlined; it must be 1 */
LONG  lOptions; /* Options */

LONG  lHits;    /* Correlation/error indicator */
```

GpiPaintRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG  lHits = GpiPaintRegion (hps, hrgn)

HPS   hps;   /* Presentation-space handle */
HRGN  hrgn;  /* Region handle */

LONG  lHits; /* Correlation/error indicator */
```

GpiPartialArc

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiPartialArc (hps, pptlCenter, fxMultiplier, fxStartAngle, fxSweepAngle)

HPS      hps;          /* Presentation-space handle */
PPOINTL  pptlCenter;   /* Center point */
FIXED    fxMultiplier; /* Multiplier */
FIXED    fxStartAngle; /* Start angle in degrees */
FIXED    fxSweepAngle; /* Sweep angle in degrees */

LONG     lHits;        /* Correlation/error indicator */
```

GpiPlayMetaFile

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

LONG   lHits = GpiPlayMetaFile (hps, hmf, lCount1, alOptarray, plSegCount, lCount2, pszDesc)

HPS    hps;        /* Presentation-space handle */
HMF    hmf;        /* Metafile handle */
LONG   lCount1;    /* Count of elements in Optarray */
PLONG  alOptarray; /* Array of options for playing */
PLONG  plSegCount; /* Reserved */
LONG   lCount2;    /* Count of bytes in Desc */
PSZ    pszDesc;    /* Descriptive record */

LONG   lHits;      /* Correlation/error indicator */
```

GpiPointArc

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiPointArc (hps, aptlPoints)

HPS      hps;        /* Presentation-space handle */
PPOINTL  aptlPoints; /* Intermediate and end points */

LONG     lHits;      /* Correlation/error indicator */
```

GpiPolyFillet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiPolyFillet (hps, lCount, aptlPoints)

HPS      hps;        /* Presentation-space handle */
LONG     lCount;     /* Number of points */
PPOINTL  aptlPoints; /* Array of points */

LONG     lHits;      /* Correlation/error indicator */
```

GpiPolyFilletSharp

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiPolyFilletSharp (hps, lCount, aptlPoints, afxSharpness)

HPS      hps;          /* Presentation-space handle */
LONG     lCount;       /* Count of points */
PPOINTL  aptlPoints;   /* An array of points */
PFIXED   afxSharpness; /* Array of sharpness values */

LONG     lHits;        /* Correlation/error indicator */
```

GpiPolyLine

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG     lHits = GpiPolyLine (hps, lCount, aptlPoints)

HPS      hps;        /* Presentation-space handle */
LONG     lCount;     /* Number of points */
PPOINTL  aptlPoints; /* Array of points */

LONG     lHits;      /* Correlation/error indicator */
```

GpiPolyMarker

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiPolyMarker (hps, lCount, aptlPoints)

HPS      hps;        /* Presentation-space handle */
LONG     lCount;     /* Number of points */
PPOINTL  aptlPoints; /* Array of points */

LONG     lHits;      /* Correlation/error indicator */
```

GpiPolySpline

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiPolySpline (hps, lCount, aptlPoints)

HPS      hps;        /* Presentation-space handle */
LONG     lCount;     /* Count of points */
PPOINTL  aptlPoints; /* An array of points */

LONG     lHits;      /* Correlation/error indicator */
```

GpiPop

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiPop (hps, lCount)

HPS   hps;      /* Presentation-space handle */
LONG  lCount;   /* Number of attributes to be restored */

BOOL  fSuccess; /* Success indicator */
```

GpiPtInRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG     lInside = GpiPtInRegion (hps, hrgn, pptlPoint)

HPS      hps;       /* Presentation-space handle */
HRGN     hrgn;      /* Region handle */
PPOINTL  pptlPoint; /* Point to be checked */

LONG     lInside;   /* Inside/error indicator */
```

GpiPtVisible

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lVisibility = GpiPtVisible (hps, pptlPoint)

HPS      hps;         /* Presentation-space handle */
PPOINTL  pptlPoint;   /* Point to be checked */

LONG     lVisibility; /* Visibility indicator */
```

GpiPutData

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

LONG   lHits = GpiPutData (hps, lFormat, plLength, pbData)

HPS    hps;       /* Presentation-space handle */
LONG   lFormat;   /* Coordinate type used */
PLONG  plLength;  /* Length of graphic data */
PBYTE  pbData;    /* Orders to be copied */

LONG   lHits;     /* Correlation/error indicator */
```

GpiQueryArcParams

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL        fSuccess = GpiQueryArcParams (hps, parcpArcParams)

HPS         hps;            /* Presentation-space handle */
PARCPARAMS  parcpArcParams; /* Arc parameters */

BOOL        fSuccess;       /* Success indicator */
```

GpiQueryAttrMode

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lMode = GpiQueryAttrMode (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lMode; /* Current attribute mode */
```

GpiQueryAttrs

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG     lDefMask = GpiQueryAttrs (hps, lPrimType, flAttrMask, ppbunAttrs)

HPS      hps;        /* Presentation-space handle */
LONG     lPrimType;  /* Primitive type */
ULONG    flAttrMask; /* Attributes mask */
PBUNDLE  ppbunAttrs; /* Attributes */

LONG     lDefMask;   /* Defaults mask */
```

GpiQueryBackColor

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lColor = GpiQueryBackColor (hps)

HPS   hps;    /* Presentation-space handle */

LONG  lColor; /* Background color */
```

GpiQueryBackMix

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lMixMode = GpiQueryBackMix (hps)

HPS   hps;      /* Presentation-space handle */

LONG  lMixMode; /* Background mix */
```

GpiQueryBitmapBits

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

LONG         lScansReturned = GpiQueryBitmapBits (hps, lScanStart, lScans, pbBuffer,
                                                  pbmiInfoTable)

HPS          hps;           /* Presentation-space handle */
LONG         lScanStart;    /* Starting line number */
LONG         lScans;        /* Number of scan lines to be returned */
PBYTE        pbBuffer;      /* Data area */
PBITMAPINFO  pbmiInfoTable; /* Bit-map information table */

LONG         lScansReturned; /* Number of scan lines actually returned */
```

GpiQueryBitmapDimension

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

BOOL      fSuccess = GpiQueryBitmapDimension (hbm, psizlBitmapDimension)

HBITMAP   hbm;                  /* Bit-map handle */
PSIZEL    psizlBitmapDimension; /* Size of bit map */

BOOL      fSuccess;             /* Success indicator */
```

GpiQueryBitmapHandle

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

HBITMAP  hbm = GpiQueryBitmapHandle (hps, lLcid)

HPS      hps;   /* Presentation-space handle */
LONG     lLcid; /* Local identifier */

HBITMAP  hbm;   /* Bit-map handle */
```

GpiQueryBitmapParameters

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

BOOL               fSuccess = GpiQueryBitmapParameters (hbm, pbmpData)

HBITMAP            hbm;      /* Bit-map handle */
PBITMAPINFOHEADER  pbmpData; /* Bit-map information header */

BOOL               fSuccess; /* Success indicator */
```

GpiQueryBoundaryData

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryBoundaryData (hps, prclBoundary)

HPS     hps;          /* Presentation-space handle */
PRECTL  prclBoundary; /* Boundary data */

BOOL    fSuccess;     /* Success indicator */
```

GpiQueryCharAngle

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL        fSuccess = GpiQueryCharAngle (hps, pgradlAngle)

HPS         hps;         /* Presentation-space handle */
PGRADIENTL  pgradlAngle; /* Baseline angle */

BOOL        fSuccess;    /* Success indicator */
```

GpiQueryCharBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryCharBox (hps, psizfxSize)

HPS     hps;        /* Presentation-space handle */
PSIZEF  psizfxSize; /* Character-box size */

BOOL    fSuccess;   /* Success indicator */
```

GpiQueryCharDirection

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lDirection = GpiQueryCharDirection (hps)

HPS   hps;        /* Presentation-space handle */

LONG  lDirection; /* Character direction */
```

GpiQueryCharMode

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lMode = GpiQueryCharMode (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lMode; /* Character mode */
```

GpiQueryCharSet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lLcid = GpiQueryCharSet (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lLcid; /* Character-set local identifier */
```

GpiQueryCharShear

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryCharShear (hps, pptlShear)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlShear; /* Character shear */

BOOL     fSuccess;  /* Success indicator */
```

GpiQueryCharStringPos

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryCharStringPos (hps, flOptions, lCount, pchString, alXincrements,
                                           aptlPositions)

HPS      hps;           /* Presentation-space handle */
ULONG    flOptions;     /* Option flag */
LONG     lCount;        /* Length of the string */
PCH      pchString;     /* The character string to be examined */
PLONG    alXincrements; /* Vector of x increment values */
PPOINTL  aptlPositions; /* Array of points */

BOOL     fSuccess;      /* Success indicator */
```

GpiQueryCharStringPosAt

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryCharStringPosAt (hps, pptlStart, flOptions, lCount, pchString,
                                             alXincrements, aptlPositions)

HPS      hps;           /* Presentation-space handle */
PPOINTL  pptlStart;     /* Starting position */
ULONG    flOptions;     /* Option flags */
LONG     lCount;        /* Length of the string */
PCH      pchString;     /* Character string to be examined */
PLONG    alXincrements; /* Vector of x increment values */
PPOINTL  aptlPositions; /* Array of points, in which the positions of each character in
                           world coordinates are returned */

BOOL     fSuccess;      /* Success indicator */
```

GpiQueryClipBox

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG    lComplexity = GpiQueryClipBox (hps, prclBound)

HPS     hps;         /* Presentation-space handle */
PRECTL  prclBound;   /* Bounding rectangle */

LONG    lComplexity; /* Complexity/error indicator */
```

GpiQueryClipRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

HRGN  hrgn = GpiQueryClipRegion (hps)

HPS   hps;  /* Presentation-space handle */

HRGN  hrgn; /* Clip-region handle (if any) */
```

GpiQueryColor

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG  lColor = GpiQueryColor (hps)

HPS   hps;    /* Presentation-space handle */

LONG  lColor; /* Color attribute */
```

GpiQueryColorData

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQueryColorData (hps, lCount, alArray)

HPS    hps;      /* Presentation-space handle */
LONG   lCount;   /* Number of elements */
PLONG  alArray;  /* Array */

BOOL   fSuccess; /* Success indicator */
```

GpiQueryColorIndex

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

LONG   lIndex = GpiQueryColorIndex (hps, flOptions, lRgbColor)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Options */
LONG   lRgbColor; /* Specifies a color in RGB terms */

LONG   lIndex;    /* Color index providing closest match to the specified color */
```

GpiQueryCp

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

USHORT  usCodePage = GpiQueryCp (hps)

HPS     hps;        /* Presentation-space handle */

USHORT  usCodePage; /* Code page */
```

GpiQueryCurrentPosition

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryCurrentPosition (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Current position */

BOOL     fSuccess;  /* Success indicator */
```

GpiQueryDefArcParams

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL        fSuccess = GpiQueryDefArcParams (hps, parcpArcParams)

HPS         hps;            /* Presentation-space handle */
PARCPARAMS  parcpArcParams; /* Default arc parameters */

BOOL        fSuccess;       /* Success indicator */
```

GpiQueryDefAttrs

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryDefAttrs (hps, lPrimType, flAttrMask, ppbunAttrs)

HPS      hps;        /* Presentation-space handle */
LONG     lPrimType;  /* Primitive type */
ULONG    flAttrMask; /* Attributes mask */
PBUNDLE  ppbunAttrs; /* Attributes */

BOOL     fSuccess;   /* Success indicator */
```

GpiQueryDefaultViewMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiQueryDefaultViewMatrix (hps, lCount, pmatlfArray)

HPS        hps;         /* Presentation-space handle */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfArray; /* Transform matrix */

BOOL       fSuccess;    /* Success indicator */
```

GpiQueryDefCharBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryDefCharBox (hps, psizlSize)

HPS     hps;       /* Presentation-space handle */
PSIZEL  psizlSize; /* Default character-box size */

BOOL    fSuccess;  /* Success indicator */
```

GpiQueryDefTag

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQueryDefTag (hps, plTag)

HPS    hps;      /* Presentation-space handle */
PLONG  plTag;    /* Default tag identifier */

BOOL   fSuccess; /* Success indicator */
```

GpiQueryDefViewingLimits

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryDefViewingLimits (hps, prclLimits)

HPS     hps;        /* Presentation-space handle */
PRECTL  prclLimits; /* Default viewing limits */

BOOL    fSuccess;   /* Success indicator */
```

GpiQueryDevice

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

HDC   hdc = GpiQueryDevice (hps)

HPS   hps;  /* Presentation-space handle */

HDC   hdc;  /* Device-context handle */
```

GpiQueryDeviceBitmapFormats

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQueryDeviceBitmapFormats (hps, lCount, alArray)

HPS    hps;      /* Presentation-space handle */
LONG   lCount;   /* Number of elements */
PLONG  alArray;  /* Data array */

BOOL   fSuccess; /* Success indicator */
```

GpiQueryDrawControl

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

LONG  lValue = GpiQueryDrawControl (hps, lControl)

HPS   hps;      /* Presentation-space handle */
LONG  lControl; /* Control whose value is to be returned */

LONG  lValue;   /* Value of the control */
```

GpiQueryDrawingMode

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

LONG  lMode = GpiQueryDrawingMode (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lMode; /* Drawing mode */
```

GpiQueryEditMode

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

LONG  lMode = GpiQueryEditMode (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lMode; /* Current editing mode */
```

GpiQueryElement

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

LONG   lRetLength = GpiQueryElement (hps, lOff, lMaxLength, pbData)

HPS    hps;        /* Presentation-space handle */
LONG   lOff;       /* Starting byte offset within the element */
LONG   lMaxLength; /* Maximum length of data that can be returned */
PBYTE  pbData;     /* Element content data */

LONG   lRetLength; /* Number of bytes returned */
```

GpiQueryElementPointer

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

LONG  lElement = GpiQueryElementPointer (hps)

HPS   hps;      /* Presentation-space handle */

LONG  lElement; /* Current element pointer */
```

GpiQueryElementType

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

LONG   lReqLength = GpiQueryElementType (hps, plType, lLength, pszData)

HPS    hps;        /* Presentation-space handle */
PLONG  plType;     /* Element type */
LONG   lLength;    /* Data length */
PSZ    pszData;    /* Description of data buffer */

LONG   lReqLength; /* Size of the data required to hold the element content */
```

GpiQueryFontFileDescriptions

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

LONG      lRemFonts = GpiQueryFontFileDescriptions (hab, pszFilename, plCount, affdescsNames)

HAB       hab;           /* Anchor-block handle */
PSZ       pszFilename;   /* Fully qualified filename */
PLONG     plCount;       /* Maximum number of family and facename pairs to be returned */
PFFDESCS  affdescsNames; /* Array of font file descriptors */

LONG      lRemFonts;     /* Returns */
```

GpiQueryFontMetrics

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL          fSuccess = GpiQueryFontMetrics (hps, lMetricsLength, pfmMetrics)

HPS           hps;           /* Presentation-space handle */
LONG          lMetricsLength; /* Length of metrics */
PFONTMETRICS  pfmMetrics;    /* Metrics of font */

BOOL          fSuccess;      /* Success indicator */
```

GpiQueryFonts

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

LONG          lRemFonts = GpiQueryFonts (hps, flOptions, pszFacename, plReqFonts,
                                         lMetricsLength, afmMetrics)

HPS           hps;            /* Presentation-space handle */
ULONG         flOptions;      /* Enumeration options */
PSZ           pszFacename;    /* Facename of fonts */
PLONG         plReqFonts;     /* Count of fonts */
LONG          lMetricsLength; /* Length of metrics */
PFONTMETRICS  afmMetrics;     /* Metrics of font */

LONG          lRemFonts;      /* Count of fonts not returned */
```

GpiQueryGraphicsField

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryGraphicsField (hps, prclField)

HPS     hps;       /* Presentation-space handle */
PRECTL  prclField; /* Graphics field */

BOOL    fSuccess;  /* Success indicator */
```

GpiQueryInitialSegmentAttrs

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

LONG  lValue = GpiQueryInitialSegmentAttrs (hps, lAttribute)

HPS   hps;        /* Presentation-space handle */
LONG  lAttribute; /* Attribute to be queried */

LONG  lValue;     /* Current initial attribute value */
```

GpiQueryKerningPairs

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

LONG           lReturned = GpiQueryKerningPairs (hps, lCount, akrnprData)

HPS            hps;        /* Presentation-space handle */
LONG           lCount;     /* The number of elements in Data */
PKERNINGPAIRS  akrnprData; /* Kerning pairs */

LONG           lReturned;  /* Number returned/error indicator */
```

GpiQueryLineEnd

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lLineEnd = GpiQueryLineEnd (hps)

HPS   hps;      /* Presentation-space handle */

LONG  lLineEnd; /* Line end */
```

GpiQueryLineJoin

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lLineJoin = GpiQueryLineJoin (hps)

HPS   hps;       /* Presentation-space handle */

LONG  lLineJoin; /* Line join */
```

GpiQueryLineType

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lLineType = GpiQueryLineType (hps)

HPS   hps;       /* Presentation-space handle */

LONG  lLineType; /* Line type */
```

GpiQueryLineWidth

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

FIXED  fxLineWidth = GpiQueryLineWidth (hps)

HPS    hps;         /* Presentation-space handle */

FIXED  fxLineWidth; /* Line width */
```

GpiQueryLineWidthGeom

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lLineWidth = GpiQueryLineWidthGeom (hps)

HPS   hps;        /* Presentation-space handle */

LONG  lLineWidth; /* Geometric line width */
```

GpiQueryLogColorTable

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

LONG   lRetCount = GpiQueryLogColorTable (hps, flOptions, lStart, lCount, alArray)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Specifies options */
LONG   lStart;    /* The starting index for which data is to be returned */
LONG   lCount;    /* Count of elements */
PLONG  alArray;   /* An array in which the information is returned */

LONG   lRetCount; /* Number of elements returned/error indicator */
```

GpiQueryMarker

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lSymbol = GpiQueryMarker (hps)

HPS   hps;     /* Presentation-space handle */

LONG  lSymbol; /* Marker symbol */
```

GpiQueryMarkerBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryMarkerBox (hps, psizfxSize)

HPS     hps;        /* Presentation-space handle */
PSIZEF  psizfxSize; /* Size of marker box */

BOOL    fSuccess;   /* Success indicator */
```

GpiQueryMarkerSet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lSet = GpiQueryMarkerSet (hps)

HPS   hps;  /* Presentation-space handle */

LONG  lSet; /* Marker set local identifier */
```

GpiQueryMetaFileBits

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQueryMetaFileBits (hmf, lOffset, lLength, pbData)

HMF    hmf;      /* Memory-metafile handle */
LONG   lOffset;  /* Byte offset */
LONG   lLength;  /* Length in bytes of the metafile data to copy */
PBYTE  pbData;   /* Metafile data */

BOOL   fSuccess; /* Success indicator */
```

GpiQueryMetaFileLength

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

LONG  lLength = GpiQueryMetaFileLength (hmf)

HMF   hmf;     /* Memory-metafile handle */

LONG  lLength; /* Total length of the metafile */
```

GpiQueryMix

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lMixMode = GpiQueryMix (hps)

HPS   hps;      /* Presentation-space handle */

LONG  lMixMode; /* Mix mode */
```

GpiQueryModelTransformMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiQueryModelTransformMatrix (hps, lCount, pmatlfArray)

HPS        hps;         /* Presentation-space handle */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfArray; /* Transform matrix */

BOOL       fSuccess;    /* Success indicator */
```

GpiQueryNearestColor

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

LONG   lRgbOut = GpiQueryNearestColor (hps, flOptions, lRgbIn)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Options */
LONG   lRgbIn;    /* Required color */

LONG   lRgbOut;   /* Nearest available color to the one specified */
```

GpiQueryNumberSetIds

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

LONG  lCount = GpiQueryNumberSetIds (hps)

HPS   hps;    /* Presentation-space handle */

LONG  lCount; /* Number of lcids */
```

GpiQueryPageViewport

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryPageViewport (hps, prclViewport)

HPS     hps;          /* Presentation-space handle */
PRECTL  prclViewport; /* Page viewport */

BOOL    fSuccess;     /* Success indicator */
```

GpiQueryPattern

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG  lPatternSymbol = GpiQueryPattern (hps)

HPS   hps;            /* Presentation-space handle */

LONG  lPatternSymbol; /* Pattern symbol */
```

GpiQueryPatternRefPoint

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryPatternRefPoint (hps, pptlRefPoint)

HPS      hps;          /* Presentation-space handle */
PPOINTL  pptlRefPoint; /* Pattern reference point */

BOOL     fSuccess;     /* Success indicator */
```

GpiQueryPatternSet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG  lSet = GpiQueryPatternSet (hps)

HPS   hps;  /* Presentation-space handle */

LONG  lSet; /* Pattern-set local identifier */
```

GpiQueryPel

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

LONG     lColor = GpiQueryPel (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Position in world coordinates */

LONG     lColor;    /* Color index of the pel */
```

GpiQueryPickAperturePosition

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryPickAperturePosition (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Pick aperture position */

BOOL     fSuccess;  /* Success indicator */
```

GpiQueryPickApertureSize

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryPickApertureSize (hps, psizlSize)

HPS     hps;       /* Presentation-space handle */
PSIZEL  psizlSize; /* Pick aperture size */

BOOL    fSuccess;  /* Success indicator */
```

GpiQueryPS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

ULONG   flOptions = GpiQueryPS (hps, psizlSize)

HPS     hps;       /* Presentation-space handle */
PSIZEL  psizlSize; /* Presentation-page size */

ULONG   flOptions; /* Presentation-space options */
```

GpiQueryRealColors

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

LONG   lRetCount = GpiQueryRealColors (hps, flOptions, lStart, lCount, alColors)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Options */
LONG   lStart;    /* Ordinal number of the first color required */
LONG   lCount;    /* Maximum number of elements */
PLONG  alColors;  /* Array in which the information is returned */

LONG   lRetCount; /* Number of elements returned */
```

GpiQueryRegionBox

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG    lComplexity = GpiQueryRegionBox (hps, hrgn, prclBound)

HPS     hps;         /* Presentation-space handle */
HRGN    hrgn;        /* Region handle */
PRECTL  prclBound;   /* Bounding rectangle */

LONG    lComplexity; /* Complexity of region/error indicator */
```

GpiQueryRegionRects

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

BOOL      fSuccess = GpiQueryRegionRects (hps, hrgn, prclBound, prgnrcControl, arclRects)

HPS       hps;           /* Presentation-space handle */
HRGN      hrgn;          /* Region handle */
PRECTL    prclBound;     /* Bounding rectangle */
PRGNRECT  prgnrcControl; /* Processing-control structure */
PRECTL    arclRects;     /* Array of rectangle structures, in which the rectangles are
                            returned */

BOOL      fSuccess;      /* Success indicator */
```

GpiQueryRGBColor

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

LONG   lRgbColor = GpiQueryRGBColor (hps, flOptions, lColorIndex)

HPS    hps;         /* Presentation-space handle */
ULONG  flOptions;   /* Options */
LONG   lColorIndex; /* Color index */

LONG   lRgbColor;   /* RGB color providing closest match to the specified color index */
```

GpiQuerySegmentAttrs

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

LONG  lValue = GpiQuerySegmentAttrs (hps, lSegid, lAttribute)

HPS   hps;        /* Presentation-space handle */
LONG  lSegid;     /* Segment identifier; must be greater than 0 */
LONG  lAttribute; /* Attribute to be queried */

LONG  lValue;     /* Current attribute value */
```

GpiQuerySegmentNames

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

LONG   lRetCount = GpiQuerySegmentNames (hps, lFirstSegid, lLastSegid, lMax, alSegids)

HPS    hps;         /* Presentation-space handle */
LONG   lFirstSegid; /* First segment in the range; must be greater than 0 */
LONG   lLastSegid;  /* Last segment in the range; must be greater than 0 */
LONG   lMax;        /* Maximum number */
PLONG  alSegids;    /* Array in which the required identifiers are returned */

LONG   lRetCount;   /* Number of identifiers returned */
```

GpiQuerySegmentPriority

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

LONG  lSegid = GpiQuerySegmentPriority (hps, lRefSegid, lOrder)

HPS   hps;       /* Presentation-space handle */
LONG  lRefSegid; /* Reference segment identifier */
LONG  lOrder;    /* Segment higher or lower */

LONG  lSegid;    /* Segment identifier */
```

GpiQuerySegmentTransformMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiQuerySegmentTransformMatrix (hps, lSegid, lCount, pmatlfArray)

HPS        hps;         /* Presentation-space handle */
LONG       lSegid;      /* Segment identifier */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfArray; /* Transform matrix */

BOOL       fSuccess;    /* Success indicator */
```

GpiQuerySetIds

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQuerySetIds (hps, lCount, alTypes, aNames, allcids)

HPS    hps;      /* Presentation-space handle */
LONG   lCount;   /* The number of objects to be queried */
PLONG  alTypes;  /* Object types */
PSTR8  aNames;   /* Font names */
PLONG  allcids;  /* Local identifiers */

BOOL   fSuccess; /* Success indicator */
```

GpiQueryStopDraw

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

LONG  lValue = GpiQueryStopDraw (hps)

HPS   hps;    /* Presentation-space handle */

LONG  lValue; /* Stop draw condition indicator */
```

GpiQueryTag

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQueryTag (hps, plTag)

HPS    hps;      /* Presentation-space handle */
PLONG  plTag;    /* Tag identifier */

BOOL   fSuccess; /* Success indicator */
```

GpiQueryTextBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiQueryTextBox (hps, lCount1, pchString, lCount2, aptlPoints)

HPS      hps;        /* Presentation-space handle */
LONG     lCount1;    /* Number of characters */
PCH      pchString;  /* The character string */
LONG     lCount2;    /* Number of points */
PPOINTL  aptlPoints; /* List of points */

BOOL     fSuccess;   /* Success indicator */
```

GpiQueryViewingLimits

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiQueryViewingLimits (hps, prclLimits)

HPS     hps;        /* Presentation-space handle */
PRECTL  prclLimits; /* Viewing limits */

BOOL    fSuccess;   /* Success indicator */
```

GpiQueryViewingTransformMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiQueryViewingTransformMatrix (hps, lCount, pmatlfArray)

HPS        hps;         /* Presentation-space handle */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfArray; /* Transform matrix */

BOOL       fSuccess;    /* Success indicator */
```

GpiQueryWidthTable

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiQueryWidthTable (hps, lFirstChar, lCount, alData)

HPS    hps;        /* Presentation-space handle */
LONG   lFirstChar; /* Codepoint of first character */
LONG   lCount;     /* Count of elements in Data */
PLONG  alData;     /* Array of width values */

BOOL   fSuccess;   /* Success indicator */
```

GpiRealizeColorTable

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiRealizeColorTable (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiRectInRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG    lInside = GpiRectInRegion (hps, hrgn, prclRect)

HPS     hps;      /* Presentation-space handle */
HRGN    hrgn;     /* Region handle */
PRECTL  prclRect; /* Test rectangle */

LONG    lInside;  /* Inside/error indicator */
```

GpiRectVisible

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

LONG    lVisibility = GpiRectVisible (hps, prclRectangle)

HPS     hps;           /* Presentation-space handle */
PRECTL  prclRectangle; /* Test rectangle, in world coordinates */

LONG    lVisibility;   /* Visibility indicator */
```

GpiRemoveDynamics

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiRemoveDynamics (hps, lFirstSegid, lLastSegid)

HPS   hps;         /* Presentation-space handle */
LONG  lFirstSegid; /* First segment in the section */
LONG  lLastSegid;  /* Last segment in the section */

BOOL  fSuccess;    /* Success indicator */
```

GpiResetBoundaryData

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiResetBoundaryData (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiResetPS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiResetPS (hps, flOptions)

HPS    hps;       /* Presentation-space handle */
ULONG  flOptions; /* Reset option */

BOOL   fSuccess;  /* Success indicator */
```

GpiRestorePS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL  fSuccess = GpiRestorePS (hps, lPSid)

HPS   hps;      /* Presentation-space handle */
LONG  lPSid;    /* Identifier of the saved presentation space that is to be restored */

BOOL  fSuccess; /* Success indicator */
```

GpiRotate

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiRotate (hps, pmatlfArray, lOptions, fxAngle, pptlCenter)

HPS        hps;         /* Presentation-space handle */
PMATRIXLF  pmatlfArray; /* Transform matrix */
LONG       lOptions;    /* Transform options */
FIXED      fxAngle;     /* Rotation angle */
PPOINTL    pptlCenter;  /* Center of rotation */

BOOL       fSuccess;    /* Success indicator */
```

GpiSaveMetaFile

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSaveMetaFile (hmf, pszFilename)

HMF   hmf;         /* Metafile handle */
PSZ   pszFilename; /* Filename */

BOOL  fSuccess;    /* Success indicator */
```

GpiSavePS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG  lPSid = GpiSavePS (hps)

HPS   hps;   /* Presentation-space handle */

LONG  lPSid; /* Identifier of saved presentation space */
```

GpiScale

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiScale (hps, pmatlfArray, lOptions, afxScale, pptlCenter)

HPS        hps;         /* Presentation-space handle */
PMATRIXLF  pmatlfArray; /* Transform matrix */
LONG       lOptions;    /* Transform options */
PFIXED     afxScale;    /* Scale factors */
PPOINTL    pptlCenter;  /* Center of scale */

BOOL       fSuccess;    /* Success indicator */
```

GpiSetArcParams

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL        fSuccess = GpiSetArcParams (hps, parcpArcParams)

HPS         hps;           /* Presentation-space handle */
PARCPARAMS  parcpArcParams; /* Arc parameters */

BOOL        fSuccess;      /* Success indicator */
```

GpiSetAttrMode

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetAttrMode (hps, lMode)

HPS   hps;      /* Presentation-space handle */
LONG  lMode;    /* Attribute mode */

BOOL  fSuccess; /* Success indicator */
```

GpiSetAttrs

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetAttrs (hps, lPrimType, flAttrMask, flDefMask, ppbunAttrs)

HPS      hps;        /* Presentation-space handle */
LONG     lPrimType;  /* Primitive type */
ULONG    flAttrMask; /* Attributes mask */
ULONG    flDefMask;  /* Defaults mask */
PBUNDLE  ppbunAttrs; /* Attributes */

BOOL     fSuccess;   /* Success indicator */
```

GpiSetBackColor

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetBackColor (hps, lColor)

HPS   hps;      /* Presentation-space handle */
LONG  lColor;   /* Background color */

BOOL  fSuccess; /* Success indicator */
```

GpiSetBackMix

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetBackMix (hps, lMixMode)

HPS   hps;      /* Presentation-space handle */
LONG  lMixMode; /* Background-mix mode */

BOOL  fSuccess; /* Success indicator */
```

GpiSetBitmap

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

HBITMAP  hbmOld = GpiSetBitmap (hps, hbm)

HPS      hps;    /* Presentation-space handle */
HBITMAP  hbm;    /* Handle of the bit map to be set */

HBITMAP  hbmOld; /* Old bit-map handle */
```

GpiSetBitmapBits

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

LONG         lScansSet = GpiSetBitmapBits (hps, lScanStart, lScans, pbBuffer, pbmiInfoTable)

HPS          hps;          /* Presentation-space handle */
LONG         lScanStart;   /* Line number */
LONG         lScans;       /* Number of scan lines to be transmitted */
PBYTE        pbBuffer;     /* Bit-map data buffer */
PBITMAPINFO  pbmiInfoTable; /* Bit-map information table */

LONG         lScansSet;    /* Number of scan lines actually set */
```

GpiSetBitmapDimension

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

BOOL      fSuccess = GpiSetBitmapDimension (hbm, psizlBitmapDimension)

HBITMAP   hbm;                  /* Bit-map handle */
PSIZEL    psizlBitmapDimension; /* Width and height of bit map */

BOOL      fSuccess;             /* Success indicator */
```

GpiSetBitmapId

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetBitmapId (hps, hbm, lLcid)

HPS      hps;      /* Presentation-space handle */
HBITMAP  hbm;      /* Bit-map handle */
LONG     lLcid;    /* Local identifier with which the bit map is to be tagged */

BOOL     fSuccess; /* Success indicator */
```

GpiSetCharAngle

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL        fSuccess = GpiSetCharAngle (hps, pgradlAngle)

HPS         hps;         /* Presentation-space handle */
PGRADIENTL  pgradlAngle; /* Baseline angle */

BOOL        fSuccess;    /* Success indicator */
```

GpiSetCharBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetCharBox (hps, psizfxBox)

HPS     hps;       /* Presentation-space handle */
PSIZEF  psizfxBox; /* Character-box size in world coordinates */

BOOL    fSuccess;  /* Success indicator */
```

GpiSetCharDirection

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetCharDirection (hps, lDirection)

HPS   hps;        /* Presentation-space handle */
LONG  lDirection; /* Character direction */

BOOL  fSuccess;   /* Success indicator */
```

GpiSetCharMode

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetCharMode (hps, lMode)

HPS   hps;      /* Presentation-space handle */
LONG  lMode;    /* Character mode */

BOOL  fSuccess; /* Success indicator */
```

GpiSetCharSet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetCharSet (hps, llcid)

HPS   hps;      /* Presentation-space handle */
LONG  llcid;    /* Character-set local identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiSetCharShear

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetCharShear (hps, pptlAngle)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlAngle; /* Character shear vector */

BOOL     fSuccess;  /* Success indicator */
```

GpiSetClipPath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetClipPath (hps, lPath, lOptions)

HPS   hps;      /* Presentation-space handle */
LONG  lPath;    /* Path control flag */
LONG  lOptions; /* Options */

BOOL  fSuccess; /* Success indicator */
```

GpiSetClipRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

LONG   lComplexity = GpiSetClipRegion (hps, hrgn, phrgnOld)

HPS    hps;         /* Presentation-space handle */
HRGN   hrgn;        /* Region handle */
PHRGN  phrgnOld;    /* Old region handle (if any) */

LONG   lComplexity; /* Complexity of clipping/error indicator */
```

GpiSetColor

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL  fSuccess = GpiSetColor (hps, lColor)

HPS   hps;      /* Presentation-space handle */
LONG  lColor;   /* Color */

BOOL  fSuccess; /* Success indicator */
```

GpiSetCp

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetCp (hps, usCodePage)

HPS     hps;        /* Presentation-space handle */
USHORT  usCodePage; /* Code-page id */

BOOL    fSuccess;   /* Success indicator */
```

GpiSetCurrentPosition

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetCurrentPosition (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* New value of current position */

BOOL     fSuccess;  /* Success indicator */
```

GpiSetDefArcParams

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL        fSuccess = GpiSetDefArcParams (hps, parcpArcParams)

HPS         hps;           /* Presentation-space handle */
PARCPARAMS  parcpArcParams; /* Default arc parameters */

BOOL        fSuccess;      /* Success indicator */
```

GpiSetDefAttrs

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetDefAttrs (hps, lPrimType, flAttrMask, ppbunAttrs)

HPS      hps;        /* Presentation-space handle */
LONG     lPrimType;  /* Primitive type */
ULONG    flAttrMask; /* Attributes mask */
PBUNDLE  ppbunAttrs; /* Default attribute values */

BOOL     fSuccess;   /* Success indicator */
```

GpiSetDefaultViewMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiSetDefaultViewMatrix (hps, lCount, pmatlfarray, lOptions)

HPS        hps;         /* Presentation-space handle */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfarray; /* Transformation matrix */
LONG       lOptions;    /* Transform options */

BOOL       fSuccess;    /* Success indicator */
```

GpiSetDefTag

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetDefTag (hps, lTag)

HPS   hps;      /* Presentation-space handle */
LONG  lTag;     /* Default tag identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiSetDefViewingLimits

```
#define INCL_GPIDEFAULTS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetDefViewingLimits (hps, prclLimits)

HPS     hps;        /* Presentation-space handle */
PRECTL  prclLimits; /* Default viewing limits */

BOOL    fSuccess;   /* Success indicator */
```

GpiSetDrawControl

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetDrawControl (hps, lControl, lValue)

HPS   hps;      /* Presentation-space handle */
LONG  lControl; /* Drawing control */
LONG  lValue;   /* Required value of the drawing control */

BOOL  fSuccess; /* Success indicator */
```

GpiSetDrawingMode

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetDrawingMode (hps, lMode)

HPS   hps;      /* Presentation-space handle */
LONG  lMode;    /* Mode to be used for subsequent drawing calls */

BOOL  fSuccess; /* Success indicator */
```

GpiSetEditMode

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetEditMode (hps, lMode)

HPS   hps;      /* Presentation-space handle */
LONG  lMode;    /* Edit mode */

BOOL  fSuccess; /* Success indicator */
```

GpiSetElementPointer

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetElementPointer (hps, lElement)

HPS   hps;      /* Presentation-space handle */
LONG  lElement; /* The element number required */

BOOL  fSuccess; /* Success indicator */
```

GpiSetElementPointerAtLabel

```
#define INCL_GPISEGEDITING   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetElementPointerAtLabel (hps, lLabel)

HPS   hps;      /* Presentation-space handle */
LONG  lLabel;   /* Required label */

BOOL  fSuccess; /* Success indicator */
```

GpiSetGraphicsField

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetGraphicsField (hps, prclField)

HPS     hps;       /* Presentation-space handle */
PRECTL  prclField; /* Graphics field */

BOOL    fSuccess;  /* Success indicator */
```

GpiSetInitialSegmentAttrs

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetInitialSegmentAttrs (hps, lAttribute, lValue)

HPS   hps;        /* Presentation-space handle */
LONG  lAttribute; /* Segment attribute */
LONG  lValue;     /* Attribute value */

BOOL  fSuccess;   /* Success indicator */
```

GpiSetLineEnd

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetLineEnd (hps, lLineEnd)

HPS   hps;      /* Presentation-space handle */
LONG  lLineEnd; /* Style of line end */

BOOL  fSuccess; /* Success indicator */
```

GpiSetLineJoin

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetLineJoin (hps, lLineJoin)

HPS   hps;       /* Presentation-space handle */
LONG  lLineJoin; /* Style of line join */

BOOL  fSuccess;  /* Success indicator */
```

GpiSetLineType

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetLineType (hps, lLineType)

HPS   hps;       /* Presentation-space handle */
LONG  lLineType; /* Line types available */

BOOL  fSuccess;  /* Success indicator */
```

GpiSetLineWidth

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiSetLineWidth (hps, fxLineWidth)

HPS    hps;         /* Presentation-space handle */
FIXED  fxLineWidth; /* Line-width multiplier */

BOOL   fSuccess;    /* Success indicator */
```

GpiSetLineWidthGeom

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetLineWidthGeom (hps, lLineWidth)

HPS   hps;        /* Presentation-space handle */
LONG  lLineWidth; /* Geometric line width */

BOOL  fSuccess;   /* Success indicator */
```

GpiSetMarker

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetMarker (hps, lSymbol)

HPS   hps;      /* Presentation-space handle */
LONG  lSymbol;  /* Marker symbol */

BOOL  fSuccess; /* Success indicator */
```

GpiSetMarkerBox

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetMarkerBox (hps, psizfxSize)

HPS     hps;        /* Presentation-space handle */
PSIZEF  psizfxSize; /* Size of marker box */

BOOL    fSuccess;   /* Success indicator */
```

GpiSetMarkerSet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetMarkerSet (hps, lSet)

HPS   hps;      /* Presentation-space handle */
LONG  lSet;     /* Marker-set local identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiSetMetaFileBits

```
#define INCL_GPIMETAFILES   /* Or use INCL_GPI or INCL_PM */

BOOL   fSuccess = GpiSetMetaFileBits (hmf, lOffset, lLength, pbBuffer)

HMF    hmf;      /* Metafile-memory handle */
LONG   lOffset;  /* Offset */
LONG   lLength;  /* Length of the metafile data */
PBYTE  pbBuffer; /* Metafile data buffer */

BOOL   fSuccess; /* Success indicator */
```

GpiSetMix

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetMix (hps, lMixMode)

HPS   hps;      /* Presentation-space handle */
LONG  lMixMode; /* Mix mode */

BOOL  fSuccess; /* Success indicator */
```

GpiSetModelTransformMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiSetModelTransformMatrix (hps, lCount, pmatlfArray, lOptions)

HPS        hps;         /* Presentation-space handle */
LONG       lCount;      /* Number of elements in matrix */
PMATRIXLF  pmatlfArray; /* Transformation matrix */
LONG       lOptions;    /* Transform options */

BOOL       fSuccess;    /* Success indicator */
```

GpiSetPageViewport

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetPageViewport (hps, prclViewport)

HPS     hps;          /* Presentation-space handle */
PRECTL  prclViewport; /* Page viewport */

BOOL    fSuccess;     /* Success indicator */
```

GpiSetPattern

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

BOOL  fSuccess = GpiSetPattern (hps, lPatternSymbol)

HPS   hps;           /* Presentation-space handle */
LONG  lPatternSymbol; /* Pattern symbol */

BOOL  fSuccess;      /* Success indicator */
```

GpiSetPatternRefPoint

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetPatternRefPoint (hps, pptlRefPoint)

HPS      hps;          /* Presentation-space handle */
PPOINTL  pptlRefPoint; /* Pattern reference point */

BOOL     fSuccess;     /* Success indicator */
```

GpiSetPatternSet

```
#define INCL_GPIPRIMITIVES   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetPatternSet (hps, lSet)

HPS   hps;      /* Presentation-space handle */
LONG  lSet;     /* Pattern-set local identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiSetPel

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM */

LONG     lHits = GpiSetPel (hps, pptlPoint)

HPS      hps;       /* Presentation-space handle */
PPOINTL  pptlPoint; /* Position in world coordinates */

LONG     lHits;     /* Correlation/error indicator */
```

GpiSetPickAperturePosition

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL     fSuccess = GpiSetPickAperturePosition (hps, pptlPick)

HPS      hps;      /* Presentation-space handle */
PPOINTL  pptlPick; /* Center of the pick aperture */

BOOL     fSuccess; /* Success indicator */
```

GpiSetPickApertureSize

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetPickApertureSize (hps, lOptions, psizlSize)

HPS     hps;       /* Presentation-space handle */
LONG    lOptions;  /* Setting option */
PSIZEL  psizlSize; /* Pick aperture size */

BOOL    fSuccess;  /* Success indicator */
```

GpiSetPS

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetPS (hps, psizlsize, flOptions)

HPS     hps;       /* Presentation-space handle */
PSIZEL  psizlsize; /* Presentation-space size */
ULONG   flOptions; /* Options */

BOOL    fSuccess;  /* Success indicator */
```

GpiSetRegion

```
#define INCL_GPIREGIONS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetRegion (hps, hrgn, lcount, arclRectangles)

HPS     hps;            /* Presentation-space handle */
HRGN    hrgn;           /* Region handle */
LONG    lcount;         /* Count of rectangles */
PRECTL  arclRectangles; /* Array of rectangles */

BOOL    fSuccess;       /* Success indicator */
```

GpiSetSegmentAttrs

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetSegmentAttrs (hps, lSegid, lAttribute, lValue)

HPS   hps;        /* Presentation-space handle */
LONG  lSegid;     /* Segment identifier */
LONG  lAttribute; /* Segment attribute */
LONG  lValue;     /* Attribute value */

BOOL  fSuccess;   /* Success indicator */
```

GpiSetSegmentPriority

```
#define INCL_GPISEGMENTS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetSegmentPriority (hps, lSegid, lRefSegid, lOrder)

HPS   hps;       /* Presentation-space handle */
LONG  lSegid;    /* Segment identifier */
LONG  lRefSegid; /* Reference segment identifier */
LONG  lOrder;    /* Segment higher or lower */

BOOL  fSuccess;  /* Success indicator */
```

GpiSetSegmentTransformMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiSetSegmentTransformMatrix (hps, lSegid, lCount, pmatlfarray,
                                                    lOptions)

HPS        hps;         /* Presentation-space handle */
LONG       lSegid;      /* Segment identifier */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfarray; /* Transformation matrix */
LONG       lOptions;    /* Transform options */

BOOL       fSuccess;    /* Success indicator */
```

GpiSetStopDraw

```
#define INCL_GPICONTROL   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetStopDraw (hps, lValue)

HPS   hps;      /* Presentation-space handle */
LONG  lValue;   /* Stop draw condition */

BOOL  fSuccess; /* Success indicator */
```

GpiSetTag

```
#define INCL_GPICORRELATION   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiSetTag (hps, lTag)

HPS   hps;      /* Presentation-space handle */
LONG  lTag;     /* Tag identifier */

BOOL  fSuccess; /* Success indicator */
```

GpiSetViewingLimits

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL    fSuccess = GpiSetViewingLimits (hps, prclLimits)

HPS     hps;        /* Presentation-space handle */
PRECTL  prclLimits; /* Viewing limits in model space */

BOOL    fSuccess;   /* Success indicator */
```

GpiSetViewingTransformMatrix

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiSetViewingTransformMatrix (hps, lCount, pmatlfArray, lOptions)

HPS        hps;         /* Presentation-space handle */
LONG       lCount;      /* Number of elements */
PMATRIXLF  pmatlfArray; /* Transformation matrix */
LONG       lOptions;    /* Transform option */

BOOL       fSuccess;    /* Success indicator */
```

GpiStrokePath

```
#define INCL_GPIPATHS   /* Or use INCL_GPI or INCL_PM */

LONG   lHits = GpiStrokePath (hps, lPath, flOptions)

HPS    hps;       /* Presentation-space handle */
LONG   lPath;     /* Identifier of path to be stroked; it must be 1 */
ULONG  flOptions; /* Stroke option */

LONG   lHits;     /* Correlation/error indicator */
```

GpiTranslate

```
#define INCL_GPITRANSFORMS   /* Or use INCL_GPI or INCL_PM */

BOOL       fSuccess = GpiTranslate (hps, pmatlfArray, lOptions, pptlTranslation)

HPS        hps;             /* Presentation-space handle */
PMATRIXLF  pmatlfArray;     /* Transform matrix */
LONG       lOptions;        /* Transform options */
PPOINTL    pptlTranslation; /* Translation */

BOOL       fSuccess;        /* Success indicator */
```

GpiUnloadFonts

```
#define INCL_GPILCIDS   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiUnloadFonts (hab, pszFilename)

HAB   hab;         /* Anchor-block handle */
PSZ   pszFilename; /* Fully qualified file name of the font resource */

BOOL  fSuccess;    /* Success indicator */
```

GpiUnrealizeColorTable

```
#define INCL_GPILOGCOLORTABLE   /* Or use INCL_GPI or INCL_PM */

BOOL  fSuccess = GpiUnrealizeColorTable (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

GpiWCBitBlt

```
#define INCL_GPIBITMAPS   /* Or use INCL_GPI or INCL_PM. Also in COMMON section */

LONG     lHits = GpiWCBitBlt (hpsTarget, hbmSource, lCount, aptlPoints, lRop, flOptions)

HPS      hpsTarget;  /* Target presentation-space handle */
HBITMAP  hbmSource;  /* Source bit-map handle */
LONG     lCount;     /* Point count */
PPOINTL  aptlPoints; /* Point array */
LONG     lRop;       /* Mixing function required */
ULONG    flOptions;  /* Options */

LONG     lHits;      /* Correlation/error indicator */
```

Chapter 5. Picture Function Calls

PicIchg

```
#define INCL_PIC   /* Or use INCL_PM */

BOOL  fSuccess = PicIchg (hab, pszSourceFileName, pszDestinationFileName, lType)

HAB   hab;                  /* Anchor-block handle */
PSZ   pszSourceFileName;    /* The name of the source file */
PSZ   pszDestinationFileName; /* The name of the destination file */
LONG  lType;                /* Specifies the type of conversion */

BOOL  fSuccess;             /* Success indicator */
```

PicPrint

```
#define INCL_PIC   /* Or use INCL_PM */

BOOL  fSuccess = PicPrint (hab, pszFilename, lType, pszParams)

HAB   hab;          /* Anchor-block handle */
PSZ   pszFilename;  /* Full path and filename of the source file */
LONG  lType;        /* Type of source file */
PSZ   pszParams;    /* Spooler parameters */

BOOL  fSuccess;     /* Success indicator */
```

Chapter 6. Profile Function Calls

PrfAddProgram

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

HPROGRAM      hprog = PrfAddProgram (hini, pprogdeDetails, hprogGroup)

HINI          hini;           /* Initialization-file handle */
PPROGDETAILS  pprogdeDetails; /* Program details for the program to be added to the program
                                 list */
HPROGRAM      hprogGroup;     /* Handle of the group to which the program is to be added */

HPROGRAM      hprog;          /* Program handle for the program added to the program list */
```

PrfChangeProgram

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

BOOL          fSuccess = PrfChangeProgram (hini, hprog, pDetails)

HINI          hini;     /* Initialization-file handle */
HPROGRAM      hprog;    /* Handle of the program whose information is to be changed */
PPROGDETAILS  pDetails; /* Program details */

BOOL          fSuccess; /* Success indicator */
```

PrfCloseProfile

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = PrfCloseProfile (hini)

HINI   hini;     /* Initialization-file handle */

BOOL   fSuccess; /* Success indicator */
```

PrfCreateGroup

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

HPROGRAM  hprogGroup = PrfCreateGroup (hini, pszTitle, chVisibility)

HINI      hini;         /* Initialization-file handle */
PSZ       pszTitle;     /* Title of the new group */
UCHAR     chVisibility; /* Visibility control */

HPROGRAM  hprogGroup;   /* Group handle for the newly-created group */
```

PrfDestroyGroup

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

BOOL      fSuccess = PrfDestroyGroup (hini, hprogGroup)

HINI      hini;       /* Initialization-file handle */
HPROGRAM  hprogGroup; /* Group handle */

BOOL      fSuccess;   /* Success indicator */
```

PrfOpenProfile

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

HINI  hini = PrfOpenProfile (hab, pszFileName)

HAB   hab;         /* Anchor-block handle */
PSZ   pszFileName; /* User-profile file name */

HINI  hini;        /* Initialization-file handle */
```

PrfQueryDefinition

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

ULONG         ulLength = PrfQueryDefinition (hini, hprog, pDetails, cchBufferMax)

HINI          hini;         /* Initialization-file handle */
HPROGRAM      hprog;        /* Handle of the program whose details are to be returned */
PPROGDETAILS  pDetails;     /* Program details */
ULONG         cchBufferMax; /* Maximum length in bytes */

ULONG         ulLength;     /* Length of returned data */
```

PrfQueryProfile

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL          fSuccess = PrfQueryProfile (hab, pprfproProfile)

HAB           hab;            /* Anchor-block handle */
PPRFPROFILE   pprfproProfile; /* Profile names structure */

BOOL          fSuccess;       /* Success indicator */
```

PrfQueryProfileData

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = PrfQueryProfileData (hini, pszApp, pszKey, pBuffer, pulBufferMax)

HINI    hini;         /* Initialization-file handle */
PSZ     pszApp;       /* Application name */
PSZ     pszKey;       /* Key name */
PVOID   pBuffer;      /* Value data */
PULONG  pulBufferMax; /* Size of value data */

BOOL    fSuccess;     /* Success indicator */
```

PrfQueryProfileInt

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

SHORT  sResult = PrfQueryProfileInt (hini, pszApp, pszKey, sDefault)

HINI   hini;     /* Initialization-file handle */
PSZ    pszApp;   /* Application name */
PSZ    pszKey;   /* Key name */
SHORT  sDefault; /* Default value */

SHORT  sResult;  /* Key value specified in the intialization file */
```

PrfQueryProfileSize

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = PrfQueryProfileSize (hini, pszApp, pszKey, pDataLen)

HINI    hini;     /* Initialization-file handle */
PSZ     pszApp;   /* Application name */
PSZ     pszKey;   /* Key name */
PULONG  pDataLen; /* Data length */

BOOL    fSuccess; /* Success indicator */
```

PrfQueryProfileString

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

ULONG  pulLength = PrfQueryProfileString (hini, pszApp, pszKey, pszDefault, pBuffer,
                                          cchBufferMax)

HINI   hini;         /* Initialization-file handle */
PSZ    pszApp;       /* Application name */
PSZ    pszKey;       /* Key name */
PSZ    pszDefault;   /* Default string */
PVOID  pBuffer;      /* Profile string */
ULONG  cchBufferMax; /* Maximum string length */

ULONG  pulLength;    /* String length returned */
```

PrfQueryProgramCategory

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

PROGCATEGORY  Category = PrfQueryProgramCategory (hini, pszExe)

HINI          hini;     /* Initialization-file handle */
PSZ           pszExe;   /* Executable-file name */

PROGCATEGORY  Category; /* Program category */
```

PrfQueryProgramHandle

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

ULONG        ulLength = PrfQueryProgramHandle (hini, pszExe, phprogArray, cchBufferMax,
                                               pulCount)

HINI         hini;         /* Initialization-file handle */
PSZ          pszExe;       /* Executable-file name */
PHPROGARRAY  phprogArray;  /* Program handles */
ULONG        cchBufferMax; /* Maximum number of program handles */
PULONG       pulCount;     /* Number of program handles returned */

ULONG        ulLength;     /* Number of bytes returned */
```

PrfQueryProgramTitles

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

ULONG       ulLength = PrfQueryProgramTitles (hini, hprogGroup, pTitles, cchBufferMax,
                                              pulCount)

HINI        hini;         /* Initialization-file handle */
HPROGRAM    hprogGroup;   /* Handle of the program or group whose information is to be
                             returned */
PPROGTITLE  pTitles;      /* Program information buffer */
ULONG       cchBufferMax; /* Buffer length */
PULONG      pulCount;     /* Number of structures returned */

ULONG       ulLength;     /* Length returned */
```

PrfRemoveProgram

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

BOOL      fSuccess = PrfRemoveProgram (hini, hprog)

HINI      hini;     /* Initialization-file handle */
HPROGRAM  hprog;    /* Program handle */

BOOL      fSuccess; /* Success indicator */
```

PrfReset

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL         fSuccess = PrfReset (hab, pprfproProfile)

HAB          hab;            /* Anchor-block handle */
PPRFPROFILE  pprfproProfile; /* Profile-names structure */

BOOL         fSuccess;       /* Success indicator */
```

PrfWriteProfileData

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = PrfWriteProfileData (hini, pszApp, pszKey, pData, cchDataLen)

HINI   hini;       /* Initialization-file handle */
PSZ    pszApp;     /* Application name */
PSZ    pszKey;     /* Key name */
PVOID  pData;      /* Value data */
ULONG  cchDataLen; /* Size of value data */

BOOL   fSuccess;   /* Success indicator */
```

PrfWriteProfileString

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = PrfWriteProfileString (hini, pszApp, pszKey, pszData)

HINI   hini;     /* Initialization-file handle */
PSZ    pszApp;   /* Application name */
PSZ    pszKey;   /* Key name */
PSZ    pszData;  /* Text string */

BOOL   fSuccess; /* Success indicator */
```

Chapter 7. Spooler Function Calls

SplQmAbort

```
#define INCL_SPL   /* Or use INCL_PM */

BOOL   fSuccess = SplQmAbort (hspl)

HSPL   hspl;     /* Spooler handle */

BOOL   fSuccess; /* Success indicator */
```

SplQmClose

```
#define INCL_SPL   /* Or use INCL_PM */

BOOL   fSuccess = SplQmClose (hspl)

HSPL   hspl;     /* Spooler handle */

BOOL   fSuccess; /* Success indicator */
```

SplQmEndDoc

```
#define INCL_SPL   /* Or use INCL_PM */

USHORT  usJob = SplQmEndDoc (hspl)

HSPL    hspl;  /* Spooler handle */

USHORT  usJob; /* Job identifier */
```

SplQmOpen

```
#define INCL_SPL   /* Or use INCL_PM */

HSPL         hspl = SplQmOpen (pszToken, lCount, pqmdopData)

PSZ          pszToken;   /* A token (nickname) that identifies spooler information */
LONG         lCount;     /* Number of items */
PQMOPENDATA  pqmdopData; /* Open parameters */

HSPL         hspl;       /* Spooler handle */
```

SplQmStartDoc

```
#define INCL_SPL   /* Or use INCL_PM */

BOOL   fSuccess = SplQmStartDoc (hspl, pszDocName)

HSPL   hspl;       /* Spooler handle */
PSZ    pszDocName; /* Document name */

BOOL   fSuccess;   /* Success indicator */
```

SplQmWrite

```
#define INCL_SPL   /* Or use INCL_PM */

BOOL   fSuccess = SplQmWrite (hspl, lCount, pbData)

HSPL   hspl;     /* Spooler handle */
LONG   lCount;   /* Length in bytes */
PBYTE  pbData;   /* Buffer of data to be written to the spool file */

BOOL   fSuccess; /* Success indicator */
```

SplQpInstall

```
#define INCL_SPL   /* Or use INCL_PM */

BOOL   fSuccess = SplQpInstall (hwnd)

HWND   hwnd;     /* Window handle */

BOOL   fSuccess; /* Success indicator */
```

SplQpQueryDt

```
#define INCL_SPL   /* Or use INCL_PM */

BOOL       fSuccess = SplQpQueryDt (plCount, aszDatatypes)

PLONG      plCount;      /* Maximum number of data types that can be returned */
PSZ *      aszDatatypes; /* An array containing the data types supported */

BOOL       fSuccess;     /* Success indicator */
```

Chapter 8. Video Function Calls

VioAssociate

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioAssociate (hdc, hvps)

HDC     hdc;     /* Device-context handle */
HVPS    hvps;    /* VIO presentation-space handle */

USHORT  usError; /* Error indicator */
```

VioCreateLogFont

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT    usError = VioCreateLogFont (pfatattrs, llcid, pName, hvps)

PFATTRS   pfatattrs;  /* Attributes of font */
LONG      llcid;      /* Local identifier */
PSTR8     pName;      /* Logical-font name */
HVPS      hvps;       /* VIO presentation-space handle */

USHORT    usError;    /* Error indicator */
```

VioCreatePS

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioCreatePS (phvps, sdepth, swidth, sFormat, sAttrs, hvpsReserved)

PHVPS   phvps;        /* VIO presentation-space handle */
SHORT   sdepth;       /* Presentation-space depth */
SHORT   swidth;       /* Presentation-space width */
SHORT   sFormat;      /* Presentation-space format */
SHORT   sAttrs;       /* Attribute bytes per character */
HVPS    hvpsReserved; /* Reserved (must be zero) */

USHORT  usError;      /* Error indicator */
```

VioDeleteSetId

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioDeleteSetId (llcid, hvps)

LONG    llcid;   /* Local identifier */
HVPS    hvps;    /* VIO presentation-space handle */

USHORT  usError; /* Error indicator */
```

VioDestroyPS

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioDestroyPS (hvps)

HVPS    hvps;    /* VIO presentation-space handle */

USHORT  usError; /* Error indicator */
```

VioGetDeviceCellSize

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT   usError = VioGetDeviceCellSize (psHeight, psWidth, hvps)

PSHORT   psHeight;  /* Current device cell height in pels */
PSHORT   psWidth;   /* Current device cell width in pels */
HVPS     hvps;      /* VIO presentation-space handle */

USHORT   usError;   /* Error indicator */
```

VioGetOrg

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT   usError = VioGetOrg (psRow, psColumn, hvps)

PSHORT   psRow;    /* Row number */
PSHORT   psColumn; /* Column number */
HVPS     hvps;     /* VIO presentation-space handle */

USHORT   usError;  /* Error indicator */
```

VioQueryFonts

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT        usError = VioQueryFonts (plRemfonts, afmMetrics, lMetricsLength, plFonts,
                                       pszFacename, flOptions, hvps)

PLONG         plRemfonts;     /* Number of fonts for which information is not returned */
PFONTMETRICS  afmMetrics;     /* Font metrics */
LONG          lMetricsLength; /* The length of each metrics record to be returned */
PLONG         plFonts;        /* Number of fonts */
PSZ           pszFacename;    /* Facename of the fonts of interest */
ULONG         flOptions;      /* Enumeration options */
HVPS          hvps;           /* VIO presentation-space handle */

USHORT        usError;        /* Error indicator */
```

VioQuerySetIds

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioQuerySetIds (allcids, pNames, alTypes, lcount, hvps)

PLONG   allcids;  /* Local identifiers */
PSTR8   pNames;   /* Font names */
PLONG   alTypes;  /* Object types */
LONG    lcount;   /* The number of objects to be queried */
HVPS    hvps;     /* VIO presentation-space handle */

USHORT  usError;  /* Error indicator */
```

VioSetDeviceCellSize

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioSetDeviceCellSize (sHeight, sWidth, hvps)

SHORT   sHeight;  /* The required device cell height in pels */
SHORT   sWidth;   /* The required device cell width in pels */
HVPS    hvps;     /* VIO presentation space handle */

USHORT  usError;  /* Error indicator */
```

VioSetOrg

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioSetOrg (sRow, sColumn, hvps)

SHORT   sRow;    /* Row number */
SHORT   sColumn; /* Column number */
HVPS    hvps;    /* VIO presentation-space handle */

USHORT  usError; /* Error indicator */
```

VioShowPS

```
#define INCL_AVIO   /* Or use INCL_PM */

USHORT  usError = VioShowPS (sDepth, sWidth, soffCell, hvps)

SHORT   sDepth;   /* The depth of the rectangle in character cell units */
SHORT   sWidth;   /* The width of the rectangle in character cell units */
SHORT   soffCell; /* Cell offset */
HVPS    hvps;     /* VIO presentation-space handle */

USHORT  usError;  /* Error indicator */
```

Chapter 9. Window Function Calls

WinAddAtom

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

ATOM       atom = WinAddAtom (hatomtblAtomTbl, pszAtomName)

HATOMTBL   hatomtblAtomTbl;  /* Atom-table handle */
PSZ        pszAtomName;      /* Atom name */

ATOM       atom;             /* Atom value */
```

WinAddProgram

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

HPROGRAM    hprogProgHandle = WinAddProgram (hab, ppibProgramInfo, hprogGroupHandle)

HAB         hab;              /* Anchor-block handle */
PPIBSTRUCT  ppibProgramInfo;  /* Program information for the program to be added to the
                                 installed program list */
HPROGRAM    hprogGroupHandle; /* Handle of the program group to which the program is to be
                                 added */

HPROGRAM    hprogProgHandle;  /* Program handle for the program added to the installed
                                 program list */
```

WinAddSwitchEntry

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HSWITCH    hswitchSwitch = WinAddSwitchEntry (pswctlSwitchData)

PSWCNTRL   pswctlSwitchData;  /* Switch data */

HSWITCH    hswitchSwitch;     /* Handle to the newly-created switch list entry */
```

WinAlarm

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fResult = WinAlarm (hwndDeskTop, fsStyle)

HWND    hwndDeskTop;  /* Desktop-window handle */
USHORT  fsStyle;      /* Alarm style */

BOOL    fResult;      /* Alarm-generated indicator */
```

WinAllocMem

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

NPBYTE  npbMem = WinAllocMem (hheapHeap, uscb)

HHEAP   hheapHeap;  /* Heap handle */
USHORT  uscb;       /* Total number of bytes to allocate */

NPBYTE  npbMem;     /* Memory pointer */
```

WinAssociateHelpInstance

```
#define INCL_WINHELP   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinAssociateHelpInstance (hwndHelpInstance, hwndApp)

HWND   hwndHelpInstance;  /* Handle of an instance of the Help Manager */
HWND   hwndApp;           /* Handle of an application window */

BOOL   fSuccess;          /* Success indicator */
```

WinAvailMem

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

USHORT  usLargest = WinAvailMem (hheapHeap, fCompact, cb)

HHEAP   hheapHeap;  /* Heap handle */
BOOL    fCompact;   /* Compact indicator */
USHORT  cb;         /* Reserved */

USHORT  usLargest;  /* Size of the largest memory block available */
```

WinBeginEnumWindows

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HENUM  Henum = WinBeginEnumWindows (hwndParent)

HWND   hwndParent;  /* Handle of the window whose child windows are to be enumerated */

HENUM  Henum;       /* Enumeration handle */
```

WinBeginPaint

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HPS     hpsPaintPS = WinBeginPaint (hwnd, hps, prclRect)

HWND    hwnd;       /* Handle of window where drawing is going to occur */
HPS     hps;        /* Presentation-space handle */
PRECTL  prclRect;   /* Bounding rectangle */

HPS     hpsPaintPS; /* Presentation-space handle */
```

WinBroadcastMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinBroadcastMsg (hwndParent, usMsgId, mpParam1, mpParam2, fsCmd)

HWND    hwndParent;  /* Parent-window handle */
USHORT  usMsgId;     /* Message identifier */
MPARAM  mpParam1;    /* Parameter 1 */
MPARAM  mpParam2;    /* Parameter 2 */
USHORT  fsCmd;       /* Broadcast message command */

BOOL    fSuccess;    /* Success indicator */
```

WinCalcFrameRect

```
#define INCL_WINFRAMEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinCalcFrameRect (hwnd, prclRect, fFrame)

HWND    hwnd;     /* Frame-window handle */
PRECTL  prclRect; /* Window rectangle */
BOOL    fFrame;   /* Frame indicator */

BOOL    fSuccess; /* Rectangle-calculated indicator */
```

WinCallMsgFilter

```
#define INCL_WINHOOKS   /* Or use INCL_WIN or INCL_PM */

BOOL    fHookRet = WinCallMsgFilter (hab, pqmsgpqmsg, usFilter)

HAB     hab;        /* Anchor-block handle */
PQMSG   pqmsgpqmsg; /* Message to be passed to the message-filter hook */
USHORT  usFilter;   /* Filter */

BOOL    fHookRet;   /* Message-filter hook return indicator */
```

WinCancelShutdown

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL  fSuccess = WinCancelShutdown (hmq, fCancelAlways)

HMQ   hmq;           /* Queue handle */
BOOL  fCancelAlways; /* Cancellation control */

BOOL  fSuccess;      /* Success indicator */
```

WinCatch

```
#define INCL_WINCATCHTHROW   /* Or use INCL_WIN or INCL_PM */

SHORT       sRetCode = WinCatch (pctchbfCatchBuf)

PCATCHBUF   pctchbfCatchBuf;  /* Saved execution environment buffer */

SHORT       sRetCode;         /* Return code */
```

WinChangeSwitchEntry

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT    usRetCode = WinChangeSwitchEntry (hswitchSwitch, pswctlSwitchData)

HSWITCH   hswitchSwitch;    /* Handle to the switch-list entry to be changed */
PSWCNTRL  pswctlSwitchData; /* Switch-control data */

USHORT    usRetCode;        /* Return code */
```

WinCloseClipbrd

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL  fSuccess = WinCloseClipbrd (hab)

HAB   hab;      /* Anchor-block handle */

BOOL  fSuccess; /* Success indicator */
```

WinCompareStrings

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

USHORT  usResult = WinCompareStrings (hab, idCodepage, idCountryCode, pszString1, pszString2,
                                      fsOptions)

HAB     hab;           /* Anchor-block handle */
USHORT  idCodepage;    /* Codepage identity of both strings */
USHORT  idCountryCode; /* Country code */
PSZ     pszString1;    /* String 1 */
PSZ     pszString2;    /* String 2 */
USHORT  fsOptions;     /* Reserved */

USHORT  usResult;      /* Comparison result */
```

WinCopyAccelTable

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

USHORT       usCopied = WinCopyAccelTable (hAccel, pacctAccelTable, usCopyMax)

HACCEL       hAccel;          /* Accelerator-table handle */
PACCELTABLE  pacctAccelTable; /* Accelerator-table data area */
USHORT       usCopyMax;       /* Maximum data area size */

USHORT       usCopied;        /* Amount copied or size required */
```

WinCopyRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinCopyRect (hab, prclDest, prclSrc)

HAB     hab;      /* Anchor-block handle */
PRECTL  prclDest; /* Destination rectangle */
PRECTL  prclSrc;  /* Source rectangle */

BOOL    fSuccess; /* Success indicator */
```

WinCpTranslateChar

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

UCHAR   ucDest = WinCpTranslateChar (hab, idCpSource, ucSource, idCpDest)

HAB     hab;        /* Anchor-block handle */
USHORT  idCpSource; /* Source-character code page */
UCHAR   ucSource;   /* Character to be translated */
USHORT  idCpDest;   /* Code page of the resultant character */

UCHAR   ucDest;     /* If nonzero, the translated character */
```

WinCpTranslateString

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinCpTranslateString (hab, idCpSource, pszSource, idCpDest, cbLenDest,
                                         pszDest)

HAB     hab;        /* Anchor-block handle */
USHORT  idCpSource; /* Source-string code page */
PSZ     pszSource;  /* String to be translated */
USHORT  idCpDest;   /* Code page of the resultant string */
USHORT  cbLenDest;  /* Maximum length of output string */
PSZ     pszDest;    /* The translated string */

BOOL    fSuccess;   /* Success indicator */
```

WinCreateAccelTable

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

HACCEL       hAccel = WinCreateAccelTable (hab, pacctAccelTable)

HAB          hab;             /* Anchor-block handle */
PACCELTABLE  pacctAccelTable; /* Accelerator table */

HACCEL       hAccel;          /* Accelerator-table handle */
```

WinCreateAtomTable

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

HATOMTBL  hatomtblAtomTbl = WinCreateAtomTable (usInitial, usBuckets)

USHORT    usInitial;       /* Initial bytes */
USHORT    usBuckets;       /* Size of the hash table */

HATOMTBL  hatomtblAtomTbl; /* Atom-table handle */
```

WinCreateCursor

```
#define INCL_WINCURSORS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinCreateCursor (hwnd, sx, sy, scx, scy, usrgf, prclClip)

HWND    hwnd;     /* Handle of window in which cursor is displayed */
SHORT   sx;       /* x position of cursor */
SHORT   sy;       /* y position of cursor */
SHORT   scx;      /* x size of cursor */
SHORT   scy;      /* y size of cursor */
USHORT  usrgf;    /* Controls the appearance of the cursor */
PRECTL  prclClip; /* Cursor rectangle */

BOOL    fSuccess; /* Success indicator */
```

WinCreateDlg

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM */

HWND          hwndDlg = WinCreateDlg (hwndParent, hwndOwner, pDlgProc, pdlgtDlgTmp,
                                      pCreateParams)

HWND          hwndParent;    /* Parent-window handle of the created dialog window */
HWND          hwndOwner;     /* Requested owner-window handle of the created dialog window
                                */
PFNWP         pDlgProc;      /* Dialog procedure for the created dialog window */
PDLGTEMPLATE  pdlgtDlgTmp;   /* Dialog template */
PVOID         pCreateParams; /* Application-defined data area */

HWND          hwndDlg;       /* Dialog-window handle */
```

WinCreateFrameControls

```
#define INCL_WINFRAMEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL         fSuccess = WinCreateFrameControls (hwndFrame, pFcdata, pszTitle)

HWND         hwndFrame;  /* Frame-window handle */
PFRAMECDATA  pFcdata;    /* Frame-control data */
PSZ          pszTitle;   /* Title string */

BOOL         fSuccess;   /* Success indicator */
```

WinCreateGroup

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

HPROGRAM  hprogGroupHandle = WinCreateGroup (hab, pszTitle, ucVisibility, flres1, flres2)

HAB       hab;              /* Anchor-block handle */
PSZ       pszTitle;         /* Title of the new group */
UCHAR     ucVisibility;     /* Visibility control */
ULONG     flres1;           /* Reserved */
ULONG     flres2;           /* Reserved */

HPROGRAM  hprogGroupHandle; /* Program-group handle for the newly-created group */
```

WinCreateHeap

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

HHEAP    hHeap = WinCreateHeap (usHeapBase, usHeapSize, usGrow, usMinDed, usMaxDed,
                                fsOptions)

USHORT   usHeapBase;  /* Selector of the segment to contain the local heap */
USHORT   usHeapSize;  /* Initial heap size in bytes */
USHORT   usGrow;      /* Heap growth size in bytes */
USHORT   usMinDed;    /* Minimum element size */
USHORT   usMaxDed;    /* Maximum element size */
USHORT   fsOptions;   /* Optional characteristics */

HHEAP    hHeap;       /* Heap handle */
```

WinCreateHelpInstance

```
#define INCL_WINHELP   /* Or use INCL_WIN or INCL_PM */

HWND       hwndhelp = WinCreateHelpInstance (hab, phinitHMInitStructure)

HAB        hab;                   /* Anchor-block handle */
PHELPINIT  phinitHMInitStructure; /* Help Manager initialization structure */

HWND       hwndhelp;              /* Help Manager handle */
```

WinCreateHelpTable

```
#define INCL_WINHELP   /* Or use INCL_WIN or INCL_PM */

BOOL        fSuccess = WinCreateHelpTable (hwndHelpInstance, phtHelpTable)

HWND        hwndHelpInstance;  /* Handle of an instance of the Help Manager */
PHELPTABLE  phtHelpTable;      /* Help table allocated by the application */

BOOL        fSuccess;          /* Success indicator */
```

WinCreateMenu

```
#define INCL_WINMENUS   /* Or use INCL_WIN or INCL_PM */

HWND   hwndMenu = WinCreateMenu (hwndOwner, pmtMenutmp)

HWND   hwndOwner;  /* Owner- and parent-window handle of the created menu window */
PVOID  pmtMenutmp; /* Menu template in binary format */

HWND   hwndMenu;   /* Menu-window handle */
```

WinCreateMsgQueue

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HMQ    hmq = WinCreateMsgQueue (hab, sQueuesize)

HAB    hab;        /* Anchor-block handle */
SHORT  sQueuesize; /* Maximum queue size */

HMQ    hmq;        /* Message-queue handle */
```

WinCreatePointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

HPOINTER  hptr = WinCreatePointer (hwndDeskTop, hbmBitMap, fPointerSize, sxHotspot,
                                   syHotspot)

HWND      hwndDeskTop;  /* Desktop-window handle or HWND_DESKTOP */
HBITMAP   hbmBitMap;    /* Bit-map handle from which the pointer image is created */
BOOL      fPointerSize; /* Pointer-size indicator */
SHORT     sxHotspot;    /* x offset of hotspot within pointer from its lower left corner (in
                           pels) */
SHORT     syHotspot;    /* y offset of hotspot within pointer from its lower left corner (in
                           pels) */

HPOINTER  hptr;         /* Pointer handle */
```

WinCreatePointerIndirect

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

HPOINTER      hptr = WinCreatePointerIndirect (hwndDeskTop, pptriPointerInfo)

HWND          hwndDeskTop;      /* Desktop-window handle or HWND_DESKTOP */
PPOINTERINFO  pptriPointerInfo; /* Pointer information structure */

HPOINTER      hptr;             /* Pointer handle */
```

WinCreateStdWindow

```
#define INCL_WINFRAMEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HWND     hwndFrame = WinCreateStdWindow (hwndParent, flStyle, pflCreateFlags, pszClassClient,
                                         pszTitle, flStyleClient, Resource, Id, phwndClient)

HWND     hwndParent;     /* Parent-window handle */
ULONG    flStyle;        /* Frame-window style */
PULONG   pflCreateFlags; /* Frame-creation flags */
PSZ      pszClassClient; /* Client-window class name */
PSZ      pszTitle;       /* Title-bar text */
ULONG    flStyleClient;  /* Client-window style */
HMODULE  Resource;       /* Resource identifier */
USHORT   Id;             /* Frame-window identifier */
PHWND    phwndClient;    /* Client-window handle */

HWND     hwndFrame;      /* Frame-window handle */
```

WinCreateSwitchEntry

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

HSWITCH   hswitchSwitch = WinCreateSwitchEntry (hab, pswctlSwitchData)

HAB       hab;              /* Anchor-block handle */
PSWCNTRL  pswctlSwitchData; /* Switch data */

HSWITCH   hswitchSwitch;    /* Handle to the newly-created switch list entry */
```

WinCreateWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND    hwnd = WinCreateWindow (hwndParent, pszClassName, pszName, flStyle, sxcoord, sycoord,
                                sWidth, sHeight, hwndOwner, hwndBehind, Id, pCtlData,
                                pPresParams)

HWND    hwndParent;   /* Parent-window handle */
PSZ     pszClassName; /* Registered-class name */
PSZ     pszName;      /* Window text */
ULONG   flStyle;      /* Window style */
SHORT   sxcoord;      /* x coordinate of window position */
SHORT   sycoord;      /* y coordinate of window position */
SHORT   sWidth;       /* Width of window, in window coordinates */
SHORT   sHeight;      /* Height of window, in window coordinates */
HWND    hwndOwner;    /* Owner-window handle */
HWND    hwndBehind;   /* Sibling-window handle */
USHORT  Id;           /* Window identifier */
PVOID   pCtlData;     /* Control data */
PVOID   pPresParams;  /* Presentation parameters */

HWND    hwnd;         /* Window handle */
```

WinDdeInitiate

```
#define INCL_WINDDE   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinDdeInitiate (hwndClient, pszAppName, pszTopicName)

HWND   hwndClient;   /* Client's window handle */
PSZ    pszAppName;   /* Application name */
PSZ    pszTopicName; /* Topic name */

BOOL   fSuccess;     /* Success indicator */
```

WinDdePostMsg

```
#define INCL_WINDDE   /* Or use INCL_WIN or INCL_PM */

BOOL        fSuccess = WinDdePostMsg (hwndTo, hwndFrom, usMsgId, pddeData, fRetry)

HWND        hwndTo;   /* Window handle of target */
HWND        hwndFrom; /* Window handle of originator */
USHORT      usMsgId;  /* Message identifier */
PDDESTRUCT  pddeData; /* DDE structure */
BOOL        fRetry;   /* Retry indicator */

BOOL        fSuccess; /* Success indicator */
```

WinDdeRespond

```
#define INCL_WINDDE   /* Or use INCL_WIN or INCL_PM */

MRESULT  mresReply = WinDdeRespond (hwndClient, hwndServer, pszAppName, pszTopicName)

HWND     hwndClient;   /* Client's window handle */
HWND     hwndServer;   /* Server's window handle */
PSZ      pszAppName;   /* Application name */
PSZ      pszTopicName; /* Topic name */

MRESULT  mresReply;    /* Message return data */
```

WinDefAVioWindowProc

```
#define INCL_AVIO   /* Or use INCL_PM */

MRESULT  mresreply = WinDefAVioWindowProc (hwnd, usMsgid, mpParam1, mpParam2)

HWND     hwnd;      /* Window handle */
USHORT   usMsgid;   /* Message identity */
MPARAM   mpParam1;  /* Parameter 1 */
MPARAM   mpParam2;  /* Parameter 2 */

MRESULT  mresreply; /* Message return data */
```

WinDefDlgProc

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

MRESULT  mresReply = WinDefDlgProc (hwndDlg, usMsgid, mpParam1, mpParam2)

HWND     hwndDlg;   /* Dialog-window handle */
USHORT   usMsgid;   /* Message identity */
MPARAM   mpParam1;  /* Parameter 1 */
MPARAM   mpParam2;  /* Parameter 2 */

MRESULT  mresReply; /* Message-return data */
```

WinDefWindowProc

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

MRESULT  mresReply = WinDefWindowProc (hwnd, usMsgid, mpParam1, mpParam2)

HWND     hwnd;      /* Window handle */
USHORT   usMsgid;   /* Message identity */
MPARAM   mpParam1;  /* Parameter 1 */
MPARAM   mpParam2;  /* Parameter 2 */

MRESULT  mresReply; /* Message-return data */
```

WinDeleteAtom

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

ATOM       ReturnCode = WinDeleteAtom (hatomtblAtomTbl, atom)

HATOMTBL   hatomtblAtomTbl;  /* Atom-table handle */
ATOM       atom;             /* Atom identifying the atom to be deleted */

ATOM       ReturnCode;       /* Return code */
```

WinDeleteLibrary

```
#define INCL_WINLOAD   /* Or use INCL_WIN or INCL_PM */

BOOL  fDeleted = WinDeleteLibrary (hab, hlibLibhandle)

HAB   hab;           /* Anchor-block handle */
HLIB  hlibLibhandle; /* Library handle to be deleted */

BOOL  fDeleted;      /* Library-deleted indicator */
```

WinDeleteProcedure

```
#define INCL_WINLOAD   /* Or use INCL_WIN or INCL_PM */

BOOL   fsuccess = WinDeleteProcedure (hab, pwndproc)

HAB    hab;      /* Anchor-block handle */
PFNWP  pwndproc; /* Window procedure identifier to be deleted */

BOOL   fsuccess; /* Procedure-deleted indicator */
```

WinDestroyAccelTable

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinDestroyAccelTable (haccelAccel)

HACCEL   haccelAccel;  /* Accelerator-table handle */

BOOL     fSuccess;     /* Success indicator */
```

WinDestroyAtomTable

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

HATOMTBL   hatomtblReturnCode = WinDestroyAtomTable (hatomtblAtomTbl)

HATOMTBL   hatomtblAtomTbl;    /* Atom-table handle */

HATOMTBL   hatomtblReturnCode; /* Return code */
```

WinDestroyCursor

```
#define INCL_WINCURSORS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinDestroyCursor (hwnd)

HWND   hwnd;      /* Window handle to which the cursor belongs */

BOOL   fSuccess; /* Success indicator */
```

WinDestroyHeap

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

HHEAP   hheapHeapRet = WinDestroyHeap (hheapHeap)

HHEAP   hheapHeap;    /* Heap handle */

HHEAP   hheapHeapRet; /* Returned heap handle */
```

WinDestroyHelpInstance

```
#define INCL_WINHELP   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinDestroyHelpInstance (hwndHelpInstance)

HWND   hwndHelpInstance;  /* Handle of the instance of the Help Manager to be destroyed */

BOOL   fSuccess;          /* Success indicator */
```

WinDestroyMsgQueue

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL  fDestroyed = WinDestroyMsgQueue (hmq)

HMQ   hmq;        /* Message-queue handle */

BOOL  fDestroyed; /* Queue-destroyed indicator */
```

WinDestroyPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL       fSuccess = WinDestroyPointer (hptrPointer)

HPOINTER   hptrPointer;  /* Handle of pointer to be destroyed */

BOOL       fSuccess;     /* Success indicator */
```

WinDestroyWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinDestroyWindow (hwnd)

HWND   hwnd;     /* Window handle */

BOOL   fSuccess; /* Window-destroyed indicator */
```

WinDismissDlg

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinDismissDlg (hwndDlg, usResult)

HWND    hwndDlg;  /* Dialog-window handle */
USHORT  usResult; /* Reply value */

BOOL    fSuccess; /* Dialog-dismissed indicator */
```

WinDispatchMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

MRESULT  mresReply = WinDispatchMsg (hab, pqmsgMsg)

HAB      hab;       /* Anchor-block handle */
PQMSG    pqmsgMsg;  /* Message structure */

MRESULT  mresReply; /* Message-return data */
```

WinDlgBox

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

USHORT   usResult = WinDlgBox (hwndParent, hwndOwner, pDlgProc, Resource, usDlgid,
                               pCreateParams)

HWND     hwndParent;    /* Parent-window handle of the created dialog window */
HWND     hwndOwner;     /* Requested owner-window handle of the created dialog window */
PFNWP    pDlgProc;      /* Dialog procedure for the created dialog window */
HMODULE  Resource;      /* Resource identity containing the dialog template */
USHORT   usDlgid;       /* Dialog-template identity within the resource file */
PVOID    pCreateParams; /* Application-defined data area */

USHORT   usResult;      /* Reply value */
```

WinDrawBitmap

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinDrawBitmap (hps, hbm, prclSrc, pptlDest, lForeColor, lBackColor,
                                   fsRgf)

HPS      hps;        /* Handle of presentation space in which the bit map is drawn */
HBITMAP  hbm;        /* Bit-map handle */
PRECTL   prclSrc;    /* Subrectangle of bit map to be drawn */
PPOINTL  pptlDest;   /* Bit-map destination */
LONG     lForeColor; /* Foreground color */
LONG     lBackColor; /* Background color */
USHORT   fsRgf;      /* Flags that determine how the bit map is drawn */

BOOL     fSuccess;   /* Success indicator */
```

WinDrawBorder

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinDrawBorder (hps, prclRectangle, sVertSideWidth, sHorizSideWidth,
                                  lBorderColor, lInteriorColor, fsCmd)

HPS     hps;            /* Presentation-space handle */
PRECTL  prclRectangle;  /* Bounding rectangle for the border */
SHORT   sVertSideWidth; /* Width of border rectangle vertical sides */
SHORT   sHorizSideWidth; /* Width of border rectangle horizontal sides */
LONG    lBorderColor;   /* Color of edge of border */
LONG    lInteriorColor; /* Color of interior of border */
USHORT  fsCmd;          /* Flags controlling the way in which the border is drawn */

BOOL    fSuccess;       /* Success indicator */
```

WinDrawPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL      fSuccess = WinDrawPointer (hps, sx, sy, hptrPointer, usHalftone)

HPS       hps;          /* Presentation-space handle into which the pointer is drawn */
SHORT     sx;           /* X coordinate at which to draw the pointer, in device coordinates
                           */
SHORT     sy;           /* Y coordinate at which to draw the pointer, in device coordinates
                           */
HPOINTER  hptrPointer;  /* Pointer handle */
USHORT    usHalftone;   /* Shading control */

BOOL      fSuccess;     /* Success indicator */
```

WinDrawText

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT   sChars = WinDrawText (hps, sCount, pchText, prclRectangle, lForeColor, lBackColor,
                              fsCmd)

HPS     hps;            /* Presentation-space handle */
SHORT   sCount;         /* Count of the number of characters in the string */
PCH     pchText;        /* Character string to be drawn */
PRECTL  prclRectangle;  /* Text rectangle */
LONG    lForeColor;     /* Foreground color */
LONG    lBackColor;     /* Background color */
USHORT  fsCmd;          /* An array of flags that determines how the text is drawn */

SHORT   sChars;         /* Count of characters drawn within the rectangle */
```

WinEmptyClipbrd

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL  fSuccess = WinEmptyClipbrd (hab)

HAB   hab;      /* Anchor-block handle */

BOOL  fSuccess; /* Success indicator */
```

WinEnablePhysInput

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

BOOL   fOldInputState = WinEnablePhysInput (hwndDeskTop, fNewInputState)

HWND   hwndDeskTop;     /* Desktop-window handle */
BOOL   fNewInputState;  /* New state for the queuing of physical input */

BOOL   fOldInputState;  /* Previous state for the queuing of physical input */
```

WinEnableWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinEnableWindow (hwnd, fNewEnabled)

HWND   hwnd;        /* Window handle */
BOOL   fNewEnabled; /* New enabled state */

BOOL   fSuccess;    /* Window enabled indicator */
```

WinEnableWindowUpdate

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinEnableWindowUpdate (hwnd, fNewVisibility)

HWND   hwnd;           /* Window handle */
BOOL   fNewVisibility; /* New visibility state */

BOOL   fSuccess;       /* Visibility-changed indicator */
```

WinEndEnumWindows

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinEndEnumWindows (henum)

HENUM   henum;    /* Enumeration handle */

BOOL    fSuccess; /* Success indicator */
```

WinEndPaint

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL  fSuccess = WinEndPaint (hps)

HPS   hps;      /* Presentation-space handle */

BOOL  fSuccess; /* Success indicator */
```

WinEnumClipbrdFmts

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

USHORT  usNext = WinEnumClipbrdFmts (hab, usPrev)

HAB     hab;    /* Anchor-block handle */
USHORT  usPrev; /* Previous clipboard-data format index */

USHORT  usNext; /* Next clipboard-data format index */
```

WinEnumDlgItem

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM */

HWND    hwndItem = WinEnumDlgItem (hwndDlg, hwnd, usCode, fLock)

HWND    hwndDlg;  /* Dialog-window handle */
HWND    hwnd;     /* Child-window handle */
USHORT  usCode;   /* Item-type code */
BOOL    fLock;    /* Lock indicator */

HWND    hwndItem; /* Item-window handle */
```

WinEqualRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fEqual = WinEqualRect (hab, prclRect1, prclRect2)

HAB     hab;       /* Anchor-block handle */
PRECTL  prclRect1; /* First rectangle */
PRECTL  prclRect2; /* Second rectangle */

BOOL    fEqual;    /* Equality indicator */
```

WinExcludeUpdateRegion

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT  sComplexity = WinExcludeUpdateRegion (hps, hwnd)

HPS    hps;         /* Presentation-space handle whose clipping region is to be updated */
HWND   hwnd;        /* Window handle */

SHORT  sComplexity; /* Complexity value */
```

WinFillRect

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinFillRect (hps, prclRect, lColor)

HPS     hps;      /* Presentation-space handle */
PRECTL  prclRect; /* Rectangle to be filled, in window coordinates */
LONG    lColor;   /* Color with which to fill the rectangle */

BOOL    fSuccess; /* Success indicator */
```

WinFindAtom

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

ATOM       atom = WinFindAtom (hatomtblAtomTbl, pszAtomName)

HATOMTBL   hatomtblAtomTbl;  /* Atom-table handle */
PSZ        pszAtomName;      /* Atom name */

ATOM       atom;             /* Atom value */
```

WinFlashWindow

```
#define INCL_WINFRAMEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinFlashWindow (hwnd, fFlash)

HWND   hwnd;     /* Handle of window to be flashed */
BOOL   fFlash;   /* Start-flashing indicator */

BOOL   fSuccess; /* Success indicator */
```

WinFocusChange

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinFocusChange (hwndDeskTop, hwndNewFocus, fsFocusChange)

HWND    hwndDeskTop;   /* Desktop-window handle */
HWND    hwndNewFocus;  /* Window handle to receive the focus */
USHORT  fsFocusChange; /* Focus changing indicators */

BOOL    fSuccess;      /* Success indicator */
```

WinFreeErrorInfo

```
#define INCL_WINERRORS   /* Or use INCL_WIN or INCL_PM */

BOOL       fSuccess = WinFreeErrorInfo (perriErrorInfo)

PERRINFO   perriErrorInfo;  /* Error-information block whose memory is to be released */

BOOL       fSuccess;        /* Success indicator */
```

WinFreeMem

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

NPBYTE  npbReturn = WinFreeMem (hheapHeap, npbMem, usLength)

HHEAP   hheapHeap;  /* Handle to a heap */
NPBYTE  npbMem;     /* Memory block to be freed */
USHORT  usLength;   /* Size of the memory to be freed */

NPBYTE  npbReturn;  /* Values as follows */
```

WinGetClipPS

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HPS     hps = WinGetClipPS (hwnd, hwndClipWindow, usClipflags)

HWND    hwnd;           /* Handle of window for which the presentation space is required */
HWND    hwndClipWindow; /* Handle of window for clipping */
USHORT  usClipflags;    /* Clipping control flags */

HPS     hps;            /* Presentation-space handle that can be used for drawing */
```

WinGetCurrentTime

```
#define INCL_WINTIMER   /* Or use INCL_WIN or INCL_PM */

ULONG  ulTime = WinGetCurrentTime (hab)

HAB    hab;    /* Anchor-block handle */

ULONG  ulTime; /* System-timer count */
```

WinGetDlgMsg

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fResult = WinGetDlgMsg (hwndDlg, pqmsgmsg)

HWND   hwndDlg;  /* Dialog-window handle */
PQMSG  pqmsgmsg; /* Message structure */

BOOL   fResult;  /* Continue message indicator */
```

WinGetErrorInfo

```
#define INCL_WINERRORS   /* Or use INCL_WIN or INCL_PM */

PERRINFO  perriErrorInfo = WinGetErrorInfo (hab)

HAB       hab;            /* Anchor-block handle */

PERRINFO  perriErrorInfo; /* Error information */
```

WinGetKeyState

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

SHORT  sKeyState = WinGetKeyState (hwndDeskTop, sVk)

HWND   hwndDeskTop;  /* Desktop-window handle: */
SHORT  sVk;          /* Virtual key value */

SHORT  sKeyState;    /* Key state */
```

WinGetLastError

```
#define INCL_WINERRORS   /* Or use INCL_WIN or INCL_PM */

ERRORID  erridErrorCode = WinGetLastError (hab)

HAB      hab;            /* Anchor-block handle */

ERRORID  erridErrorCode; /* Last-error state */
```

WinGetMinPosition

```
#define INCL_WINFRAMEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinGetMinPosition (hwnd, pSwp, pptlPoint)

HWND     hwnd;      /* Frame-window handle */
PSWP     pSwp;      /* Set window position structure */
PPOINTL  pptlPoint; /* Preferred position */

BOOL     fSuccess;  /* Success indicator */
```

WinGetMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fResult = WinGetMsg (hab, pqmsgmsg, hwndFilter, usFirst, usLast)

HAB     hab;        /* Anchor-block handle */
PQMSG   pqmsgmsg;   /* Message structure */
HWND    hwndFilter; /* Window filter */
USHORT  usFirst;    /* First message identity */
USHORT  usLast;     /* Last message identity */

BOOL    fResult;    /* Continue message indicator */
```

WinGetNextWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND    hwndNext = WinGetNextWindow (henum)

HENUM   henum;    /* Enumeration handle */

HWND    hwndNext; /* Next window handle in enumeration list */
```

WinGetPhysKeyState

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

SHORT  sKeyState = WinGetPhysKeyState (hwndDeskTop, sScancode)

HWND   hwndDeskTop;  /* Desktop-window handle */
SHORT  sScancode;    /* Hardware scancode */

SHORT  sKeyState;    /* Key state */
```

WinGetPS

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HPS    hps = WinGetPS (hwnd)

HWND   hwnd;  /* Handle of window for which the presentation space is required */

HPS    hps;   /* Presentation-space handle that can be used for drawing in the window */
```

WinGetScreenPS

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HPS    hpsScreenPS = WinGetScreenPS (hwndDeskTop)

HWND   hwndDeskTop;  /* Desktop-window handle */

HPS    hpsScreenPS;  /* Presentation-space handle */
```

WinGetSysBitmap

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

HBITMAP  hbm = WinGetSysBitmap (hwndDeskTop, iIndex)

HWND     hwndDeskTop;  /* Desktop-window handle */
USHORT   iIndex;       /* System bit-map index value */

HBITMAP  hbm;          /* System bit-map handle */
```

WinInflateRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinInflateRect (hab, prclrect, scx, scy)

HAB     hab;      /* Anchor-block handle */
PRECTL  prclrect; /* Rectangle to be expanded */
SHORT   scx;      /* Horizontal expansion */
SHORT   scy;      /* Vertical expansion */

BOOL    fSuccess; /* Success indicator */
```

WinInitialize

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HAB      hab = WinInitialize (fsOptions)

USHORT   fsOptions;  /* Initialization options */

HAB      hab;        /* Anchor-block handle */
```

WinInSendMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL  fProcessing = WinInSendMsg (hab)

HAB   hab;         /* Anchor-block handle */

BOOL  fProcessing; /* Message-processing indicator */
```

WinInstStartApp

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HAPP       happ = WinInstStartApp (hini, hwndNotifyWindow, cCount, aszApplication,
                                   pszCmdLine, pData, fsOptions)

HINI       hini;             /* Initialization-file handle */
HWND       hwndNotifyWindow; /* Notification-window handle */
USHORT     cCount;           /* Count of elements in the Application parameter. Must be 1 or
                                2 */
PSZ FAR *  aszApplication;   /* Identifier of the application to be started */
PSZ        pszCmdLine;       /* Input parameters for the application to be started */
PVOID      pData;            /* Start data */
USHORT     fsOptions;        /* Option indicators */

HAPP       happ;             /* Application handle */
```

WinIntersectRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinIntersectRect (hab, prclDest, prclRect1, prclRect2)

HAB     hab;       /* Anchor-block handle */
PRECTL  prclDest;  /* Intersection rectangle */
PRECTL  prclRect1; /* First rectangle */
PRECTL  prclRect2; /* Second rectangle */

BOOL    fSuccess;  /* Success indicator */
```

WinInvalidateRect

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinInvalidateRect (hwnd, prclPrc, fIncludeClippedChildren)

HWND    hwnd;                   /* Handle of window whose update region is to be changed */
PRECTL  prclPrc;                /* Update rectangle */
BOOL    fIncludeClippedChildren; /* Invalidation-scope indicator */

BOOL    fSuccess;               /* Success indicator */
```

WinInvalidateRegion

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinInvalidateRegion (hwnd, hrgn, fIncludeClippedChildren)

HWND   hwnd;                   /* Handle of window whose update region is to be changed */
HRGN   hrgn;                   /* Handle of the region to be added to the window's update
                                  region */
BOOL   fIncludeClippedChildren; /* Invalidation-scope indicator */

BOOL   fSuccess;               /* Success indicator */
```

WinInvertRect

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinInvertRect (hps, prclRect)

HPS     hps;      /* Presentation-space handle */
PRECTL  prclRect; /* Rectangle to be inverted */

BOOL    fSuccess; /* Success indicator */
```

WinIsChild

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fRelated = WinIsChild (hwndChild, hwndParent)

HWND   hwndChild;  /* Child-window handle */
HWND   hwndParent; /* Parent-window handle */

BOOL   fRelated;   /* Related indicator */
```

WinIsRectEmpty

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fEmpty = WinIsRectEmpty (hab, prclprc)

HAB     hab;     /* Anchor-block handle */
PRECTL  prclprc; /* Rectangle */

BOOL    fEmpty;  /* Empty indicator */
```

WinIsThreadActive

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL  fActive = WinIsThreadActive (hab)

HAB   hab;      /* Anchor-block handle of calling thread */

BOOL  fActive; /* Active-window indicator */
```

WinIsWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL  fValid = WinIsWindow (hab, hwnd)

HAB   hab;    /* Anchor-block handle */
HWND  hwnd;   /* Window handle */

BOOL  fValid; /* Validity indicator */
```

WinIsWindowEnabled

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fEnabled = WinIsWindowEnabled (hwnd)

HWND   hwnd;     /* Window handle */

BOOL   fEnabled; /* Enabled-state indicator */
```

WinIsWindowShowing

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fShowing = WinIsWindowShowing (hwnd)

HWND   hwnd;     /* Window handle */

BOOL   fShowing; /* Showing state indicator */
```

WinIsWindowVisible

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fVisible = WinIsWindowVisible (hwnd)

HWND   hwnd;     /* Window handle */

BOOL   fVisible; /* Visibility-state indicator */
```

WinLoadAccelTable

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

HACCEL   haccelAccel = WinLoadAccelTable (hab, Resource, idAccelTable)

HAB      hab;          /* Anchor-block handle */
HMODULE  Resource;     /* Resource identity containing the accelerator table */
USHORT   idAccelTable; /* Accelerator-table identifier, within the resource file */

HACCEL   haccelAccel;  /* Accelerator-table handle */
```

WinLoadDlg

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HWND     hwndDlg = WinLoadDlg (hwndParent, hwndOwner, pDlgProc, Resource, idDlgid,
                               pCreateParams)

HWND     hwndParent;     /* Parent-window handle of the created dialog window */
HWND     hwndOwner;      /* Requested owner-window handle of the created dialog window */
PFNWP    pDlgProc;       /* Dialog procedure for the created dialog window */
HMODULE  Resource;       /* Resource identity containing the dialog template */
USHORT   idDlgid;        /* Dialog-template identity within the resource file */
PVOID    pCreateParams; /* Application-defined data area */

HWND     hwndDlg;        /* Dialog-window handle */
```

WinLoadHelpTable

```
#define INCL_WINHELP   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinLoadHelpTable (hwndHelpInstance, idHelpTable, Module)

HWND     hwndHelpInstance;  /* Handle of a Help Manager instance */
USHORT   idHelpTable;       /* Identity of the help table */
HMODULE  Module;            /* Handle of the module which contains the help table and help
                               subtable resources */

BOOL     fSuccess;          /* Success indicator */
```

WinLoadLibrary

```
#define INCL_WINLOAD   /* Or use INCL_WIN or INCL_PM */

HLIB  hlibLibhandle = WinLoadLibrary (hab, pszLibname)

HAB   hab;           /* Anchor-block handle */
PSZ   pszLibname;    /* Library name */

HLIB  hlibLibhandle; /* Library handle */
```

WinLoadMenu

```
#define INCL_WINMENUS   /* Or use INCL_WIN or INCL_PM */

HWND     hwndMenu = WinLoadMenu (hwndOwner, Resource, idMenuid)

HWND     hwndOwner;  /* Owner- and parent-window handle */
HMODULE  Resource;   /* Resource identifier */
USHORT   idMenuid;   /* Menu identifier within the resource file */

HWND     hwndMenu;   /* Menu-window handle */
```

WinLoadPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

HPOINTER  hptr = WinLoadPointer (hwndDeskTop, Resource, idPointer)

HWND      hwndDeskTop;  /* Desktop-window handle */
HMODULE   Resource;     /* Resource identity containing the pointer definition */
USHORT    idPointer;    /* Identifier of the pointer to be loaded */

HPOINTER  hptr;         /* Pointer handle */
```

WinLoadProcedure

```
#define INCL_WINLOAD   /* Or use INCL_WIN or INCL_PM */

PFNWP  pWndproc = WinLoadProcedure (hab, hlibLibhandle, pszProcname)

HAB    hab;           /* Anchor-block handle */
HLIB   hlibLibhandle; /* Library handle */
PSZ    pszProcname;   /* Procedure name */

PFNWP  pWndproc;      /* Window-procedure identifier */
```

WinLoadString

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT    sLength = WinLoadString (hab, Resource, idString, sBufferMax, pszBuffer)

HAB      hab;         /* Anchor-block handle */
HMODULE  Resource;    /* Resource identity containing the string */
USHORT   idString;    /* String identifier */
SHORT    sBufferMax;  /* Size of buffer */
PSZ      pszBuffer;   /* Buffer that is to receive the string */

SHORT    sLength;     /* The length of the string returned */
```

WinLockHeap

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

PVOID   pStart = WinLockHeap (hheapHeap)

HHEAP   hheapHeap;  /* Handle to a heap */

PVOID   pStart;     /* Segment containing the passed heap */
```

WinLockVisRegions

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinLockVisRegions (hwndDeskTop, fLock)

HWND   hwndDeskTop;  /* Desktop-window handle or HWND_DESKTOP */
BOOL   fLock;        /* Indicates whether the visible regions are being locked or unlocked
                        */

BOOL   fSuccess;     /* Success indicator */
```

WinLockWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND   hwndRet = WinLockWindow (hwnd, fLock)

HWND   hwnd;    /* Window handle */
BOOL   fLock;   /* Lock indicator */

HWND   hwndRet; /* Locked-window handle */
```

WinLockWindowUpdate

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinLockWindowUpdate (hwndDeskTop, hwndLockUpdate)

HWND   hwndDeskTop;    /* Desktop handle of the screen containing the window to be locked */
HWND   hwndLockUpdate; /* Handle of window in which output is to be prevented */

BOOL   fSuccess;       /* Success indicator */
```

WinMakePoints

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinMakePoints (hab, pwptppt, ccount)

HAB      hab;      /* Anchor-block handle */
PWPOINT  pwptppt;  /* Points to be converted */
USHORT   ccount;   /* Number of points to be converted */

BOOL     fSuccess; /* Success indicator */
```

WinMakeRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinMakeRect (hab, pwrcprc)

HAB     hab;      /* Anchor-block handle */
PWRECT  pwrcprc;  /* Rectangle to be converted */

BOOL    fSuccess; /* Success indicator */
```

WinMapDlgPoints

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinMapDlgPoints (hwndDlg, aptlPoints, usCount, fOptions)

HWND     hwndDlg;    /* Dialog-window handle */
PPOINTL  aptlPoints; /* Coordinate points to be mapped */
USHORT   usCount;    /* Number of coordinate points */
BOOL     fOptions;   /* Calculation control */

BOOL     fSuccess;   /* Coordinates-mapped indicator */
```

WinMapWindowPoints

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinMapWindowPoints (hwndFrom, hwndTo, aptlPoints, sCount)

HWND     hwndFrom;   /* Handle of the window from whose coordinates points are to be mapped
                        */
HWND     hwndTo;     /* Handle of the window to whose coordinates points are to be mapped */
PPOINTL  aptlPoints; /* Points to be mapped to the new coordinate system */
SHORT    sCount;     /* Number of points to be mapped */

BOOL     fSuccess;   /* Success indicator */
```

WinMessageBox

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

USHORT  usResponse = WinMessageBox (hwndParent, hwndOwner, pszText, pszTitle, usWindow,
                                    fsStyle)

HWND    hwndParent;  /* Parent-window handle of the created message-box window */
HWND    hwndOwner;   /* Requested owner-window handle of the created message-box window */
PSZ     pszText;     /* Message-box window message */
PSZ     pszTitle;    /* Message-box window title */
USHORT  usWindow;    /* Message-box window identity */
USHORT  fsStyle;     /* Message-box window style */

USHORT  usResponse;  /* User-response value */
```

WinMsgMuxSemWait

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

USHORT    usrc = WinMsgMuxSemWait (pusIndexNbr, pList, dtTimeout)

PUSHORT   pusIndexNbr;  /* Index of the cleared semaphore in the list */
PVOID     pList;        /* List of semaphore descriptors */
LONG      dtTimeout;    /* Function time-out value */

USHORT    usrc;         /* Return code */
```

WinMsgSemWait

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

USHORT  usrc = WinMsgSemWait (hsem, dtTimeout)

HSEM    hsem;      /* Semaphore handle */
LONG    dtTimeout; /* Function time-out value */

USHORT  usrc;      /* Return code */
```

WinMultWindowFromIDs

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT   sWindows = WinMultWindowFromIDs (hwndParent, ahwnd, usFirst, usLast)

HWND    hwndParent;  /* Parent-window handle */
PHWND   ahwnd;       /* Window handles */
USHORT  usFirst;     /* First window identity value in the range (inclusive) */
USHORT  usLast;      /* Last window identity value in the range (inclusive) */

SHORT   sWindows;    /* Number of window handles returned */
```

WinNextChar

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

PSZ     pszNextChar = WinNextChar (hab, usCodepage, usCountry, pszCurrentChar)

HAB     hab;            /* Anchor-block handle */
USHORT  usCodepage;     /* Code page */
USHORT  usCountry;      /* Country code */
PSZ     pszCurrentChar; /* Current character in a null-terminated string */

PSZ     pszNextChar;    /* Next character in the null-terminated string */
```

WinOffsetRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinOffsetRect (hab, prclrect, scx, scy)

HAB     hab;      /* Anchor-block handle */
PRECTL  prclrect; /* Rectangle to be offset */
SHORT   scx;      /* x value of offset */
SHORT   scy;      /* y value of offset */

BOOL    fSuccess; /* Success indicator */
```

WinOpenClipbrd

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL  fSuccess = WinOpenClipbrd (hab)

HAB   hab;      /* Anchor-block handle */

BOOL  fSuccess; /* Success indicator */
```

WinOpenWindowDC

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HDC    hdc = WinOpenWindowDC (hwnd)

HWND   hwnd;  /* Window handle */

HDC    hdc;   /* Device-context handle */
```

WinPeekMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fResult = WinPeekMsg (hab, pqmsgmsg, hwndFilter, usFirst, usLast, fsOptions)

HAB     hab;        /* Anchor-block handle */
PQMSG   pqmsgmsg;   /* Message structure */
HWND    hwndFilter; /* Window filter */
USHORT  usFirst;    /* First message identity */
USHORT  usLast;     /* Last message identity */
USHORT  fsOptions;  /* Options */

BOOL    fResult;    /* Message-available indicator */
```

WinPostMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fResult = WinPostMsg (hwnd, usMsgid, mpParam1, mpParam2)

HWND    hwnd;     /* Window handle */
USHORT  usMsgid;  /* Message identity */
MPARAM  mpParam1; /* Parameter 1 */
MPARAM  mpParam2; /* Parameter 2 */

BOOL    fResult;  /* Message-posted indicator */
```

WinPostQueueMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinPostQueueMsg (hmq, usMsgId, mpParam1, mpParam2)

HMQ     hmq;      /* Message-queue handle */
USHORT  usMsgId;  /* Message identifier */
MPARAM  mpParam1; /* Parameter 1 */
MPARAM  mpParam2; /* Parameter 2 */

BOOL    fSuccess; /* Success indicator */
```

WinPrevChar

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

PSZ     pszPrevChar = WinPrevChar (hab, usCodepage, usCountry, pszStart, pszCurrentChar)

HAB     hab;            /* Anchor-block handle */
USHORT  usCodepage;     /* Code page */
USHORT  usCountry;      /* Country code */
PSZ     pszStart;       /* Character string that contains CurrentChar */
PSZ     pszCurrentChar; /* Current character */

PSZ     pszPrevChar;    /* Previous character */
```

WinProcessDlg

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM */

USHORT  usResult = WinProcessDlg (hwndDlg)

HWND    hwndDlg;  /* Dialog-window handle */

USHORT  usResult; /* Reply value */
```

WinPtInRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL     f = WinPtInRect (hab, prclrect, pptlpoint)

HAB      hab;       /* Anchor-block handle */
PRECTL   prclrect;  /* Rectangle to be queried */
PPOINTL  pptlpoint; /* Point to be queried */

BOOL     f;         /* Success indicator */
```

WinQueryAccelTable

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

HACCEL  haccelAccel = WinQueryAccelTable (hab, hwndFrame)

HAB     hab;         /* Anchor-block handle */
HWND    hwndFrame;   /* Frame-window handle */

HACCEL  haccelAccel; /* Accelerator-table handle */
```

WinQueryActiveWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND   hwndActive = WinQueryActiveWindow (hwndParent, fLock)

HWND   hwndParent;  /* Parent-window handle for which the active window is required */
BOOL   fLock;       /* Windows lock-state indicator */

HWND   hwndActive;  /* Active-window handle */
```

WinQueryAnchorBlock

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

HAB    hab = WinQueryAnchorBlock (hwnd)

HWND   hwnd;  /* Window handle */

HAB    hab;   /* Anchor block handle */
```

WinQueryAtomLength

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

USHORT     usretlen = WinQueryAtomLength (hatomtblAtomTbl, atom)

HATOMTBL   hatomtblAtomTbl;  /* Atom-table handle */
ATOM       atom;             /* Atom whose associated character-string length is to be
                                returned */

USHORT     usretlen;         /* String length */
```

WinQueryAtomName

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

USHORT     usretlen = WinQueryAtomName (hatomtblAtomTbl, atom, pszBuffer, usBufferMax)

HATOMTBL   hatomtblAtomTbl;  /* Atom-table handle */
ATOM       atom;             /* Atom; identifies the character string to be retrieved */
PSZ        pszBuffer;        /* Buffer to receive the character string */
USHORT     usBufferMax;      /* Buffer size in bytes */

USHORT     usretlen;         /* Length of retrieved character string */
```

WinQueryAtomUsage

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

USHORT     uscount = WinQueryAtomUsage (hatomtblAtomTbl, atom)

HATOMTBL   hatomtblAtomTbl;  /* Atom-table handle */
ATOM       atom;             /* Atom whose use count is to be returned */

USHORT     uscount;          /* Use count of the atom */
```

WinQueryCapture

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

HWND   hwnd = WinQueryCapture (hwndDesktop, fLock)

HWND   hwndDesktop;  /* Desktop-window handle */
BOOL   fLock;        /* Lock-request indicator */

HWND   hwnd;         /* Handle of the window with the pointer captured */
```

WinQueryClassInfo

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL        fExists = WinQueryClassInfo (hab, pszClassName, pclsiClassInfo)

HAB         hab;            /* Anchor-block handle */
PSZ         pszClassName;   /* Class name */
PCLASSINFO  pclsiClassInfo; /* Class information structure */

BOOL        fExists;        /* Class-exists indicator */
```

WinQueryClassName

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT  sRetLen = WinQueryClassName (hwnd, sLength, pchBuffer)

HWND   hwnd;      /* Window handle */
SHORT  sLength;   /* Length of Buffer */
PCH    pchBuffer; /* Class name */

SHORT  sRetLen;   /* Returned class name length */
```

WinQueryClipbrdData

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

ULONG   ulData = WinQueryClipbrdData (hab, usfmt)

HAB     hab;    /* Anchor-block handle */
USHORT  usfmt;  /* Format of the data to be accessed */

ULONG   ulData; /* Handle to the clipboard data */
```

WinQueryClipbrdFmtInfo

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL     fExists = WinQueryClipbrdFmtInfo (hab, usfmt, pusFmtInfo)

HAB      hab;        /* Anchor-block handle */
USHORT   usfmt;      /* Format of the data to be queried */
PUSHORT  pusFmtInfo; /* Memory model and usage flags */

BOOL     fExists;    /* Format-exists indicator */
```

WinQueryClipbrdOwner

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

HWND  hwndClipbrdOwner = WinQueryClipbrdOwner (hab, fLock)

HAB   hab;              /* Anchor-block handle */
BOOL  fLock;            /* Lock state indicator of clipboard owner window */

HWND  hwndClipbrdOwner; /* Window handle of the current clipboard owner */
```

WinQueryClipbrdViewer

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

HWND  hwndClipbrdViewer = WinQueryClipbrdViewer (hab, fLock)

HAB   hab;               /* Anchor-block handle */
BOOL  fLock;             /* Lock-state indicator of clipboard viewer window */

HWND  hwndClipbrdViewer; /* Current clipboard viewer window handle */
```

WinQueryCp

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

USHORT  usCodePage = WinQueryCp (hmq)

HMQ     hmq;        /* Message queue */

USHORT  usCodePage; /* Code page */
```

WinQueryCpList

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

USHORT   usTotCount = WinQueryCpList (hab, uscount, ausCodepage)

HAB      hab;         /* Anchor-block handle */
USHORT   uscount;     /* Maximum number of code pages returned */
PUSHORT  ausCodepage; /* Code page list */

USHORT   usTotCount;  /* Total number of code pages available */
```

WinQueryCursorInfo

```
#define INCL_WINCURSORS   /* Or use INCL_WIN or INCL_PM */

BOOL         fCursor = WinQueryCursorInfo (hwndDeskTop, pcsriCursorInfo)

HWND         hwndDeskTop;     /* Desktop-window handle */
PCURSORINFO  pcsriCursorInfo; /* Cursor information */

BOOL         fCursor;         /* Current-cursor indicator */
```

WinQueryDefinition

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

USHORT      usLength = WinQueryDefinition (hab, hprogProgHandle, ppibProgramInfo,
                                           usMaxLength)

HAB         hab;             /* Anchor-block handle */
HPROGRAM    hprogProgHandle; /* Handle of the program whose information is to be returned */
PPIBSTRUCT  ppibProgramInfo; /* Program-information data */
USHORT      usMaxLength;     /* Maximum length in bytes */

USHORT      usLength;        /* Length of returned data */
```

WinQueryDesktopWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND  hwndDeskTop = WinQueryDesktopWindow (hab, hdc)

HAB   hab;          /* Anchor-block handle */
HDC   hdc;          /* Device-context handle */

HWND  hwndDeskTop; /* Desktop-window handle */
```

WinQueryDlgItemShort

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinQueryDlgItemShort (hwndDlg, idItem, psResult, fSigned)

HWND    hwndDlg;  /* Parent-window handle */
USHORT  idItem;   /* Identity of the child window whose text is to be converted */
PSHORT  psResult; /* Integer value resulting from the conversion */
BOOL    fSigned;  /* Sign indicator */

BOOL    fSuccess; /* Success indicator */
```

WinQueryDlgItemText

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

USHORT  usRetLen = WinQueryDlgItemText (hwndDlg, idItem, sMaxText, pszText)

HWND    hwndDlg;  /* Parent-window handle */
USHORT  idItem;   /* Identity of the child window whose text is to be queried */
SHORT   sMaxText; /* Length of Text */
PSZ     pszText;  /* Output string */

USHORT  usRetLen; /* Actual number of characters returned */
```

WinQueryDlgItemTextLength

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

SHORT   sRetLen = WinQueryDlgItemTextLength (hwndDlg, idItem)

HWND    hwndDlg;  /* Parent-window handle */
USHORT  idItem;   /* Identity of the child window whose text is to be queried */

SHORT   sRetLen;  /* Length of text */
```

WinQueryFocus

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

HWND   hwndFocus = WinQueryFocus (hwndDeskTop, fLock)

HWND   hwndDeskTop;  /* Desktop-window handle */
BOOL   fLock;        /* Window lock state indicator */

HWND   hwndFocus;    /* Focus-handle */
```

WinQueryHelpInstance

```
#define INCL_WINHELP   /* Or use INCL_WIN or INCL_PM */

HWND   hwndHelp = WinQueryHelpInstance (hwndApp)

HWND   hwndApp;  /* Handle of the application window */

HWND   hwndHelp; /* Help Manager window handle */
```

WinQueryMsgPos

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinQueryMsgPos (hab, pptlptrpos)

HAB      hab;        /* Anchor-block handle */
PPOINTL  pptlptrpos; /* Pointer position in screen coordinates */

BOOL     fSuccess;   /* Success indicator */
```

WinQueryMsgTime

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

ULONG  ulTime = WinQueryMsgTime (hab)

HAB    hab;    /* Anchor-block handle */

ULONG  ulTime; /* Time in milliseconds */
```

WinQueryObjectWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND   hwndObject = WinQueryObjectWindow (hwndDeskTop)

HWND   hwndDeskTop;  /* Desktop-window handle */

HWND   hwndObject;   /* Object-window handle */
```

WinQueryPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

HPOINTER  hptrPointer = WinQueryPointer (hwndDeskTop)

HWND      hwndDeskTop;  /* Desktop-window handle */

HPOINTER  hptrPointer;  /* Pointer handle */
```

WinQueryPointerInfo

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL          fSuccess = WinQueryPointerInfo (hptr, pptriPointerInfo)

HPOINTER      hptr;             /* Pointer handle */
PPOINTERINFO  pptriPointerInfo; /* Pointer-information structure */

BOOL          fSuccess;         /* Success indicator */
```

WinQueryPointerPos

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinQueryPointerPos (hwndDeskTop, pptlPoint)

HWND     hwndDeskTop;  /* Desktop-window handle */
PPOINTL  pptlPoint;    /* Pointer position in screen coordinates */

BOOL     fSuccess;     /* Pointer position returned indicator */
```

WinQueryPresParam

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

ULONG   cbRetLen = WinQueryPresParam (hwnd, idAttrType1, idAttrType2, pidAttrTypeFound,
                                      cbAttrValueLen, pAttrValue, fsOptions)

HWND    hwnd;             /* Window handle */
ULONG   idAttrType1;      /* First attribute type identity */
ULONG   idAttrType2;      /* Second attribute type identity */
PULONG  pidAttrTypeFound; /* Attribute type identity found */
ULONG   cbAttrValueLen;   /* Byte count of the size of the AttrValue parameter */
PVOID   pAttrValue;       /* Attribute value */
USHORT  fsOptions;        /* Options */

ULONG   cbRetLen;         /* Length of presentation parameter value passed back */
```

WinQueryProfileData

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinQueryProfileData (hab, pszAppName, pszKeyName, pValue, pusSize)

HAB      hab;        /* Anchor-block handle */
PSZ      pszAppName; /* Application name */
PSZ      pszKeyName; /* Key name */
PVOID    pValue;     /* Value data */
PUSHORT  pusSize;    /* Size of value data */

BOOL     fSuccess;   /* Success indicator */
```

WinQueryProfileInt

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

SHORT  sRequiredInteger = WinQueryProfileInt (hab, pszAppName, pszKeyName, sDefault)

HAB    hab;              /* Anchor-block handle */
PSZ    pszAppName;       /* Application name */
PSZ    pszKeyName;       /* Key name */
SHORT  sDefault;         /* Default value */

SHORT  sRequiredInteger; /* Key value specified in the intialization file */
```

WinQueryProfileSize

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

USHORT   usRetCode = WinQueryProfileSize (hab, pszAppName, pszKeyName, pusValue)

HAB      hab;        /* Anchor-block handle */
PSZ      pszAppName; /* Application name */
PSZ      pszKeyName; /* Key name */
PUSHORT  pusValue;   /* Data length */

USHORT   usRetCode;  /* Success indicator */
```

WinQueryProfileString

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

USHORT  usPstring = WinQueryProfileString (hab, pszAppName, pszKeyName, pszDefault,
                                           pProfileString, usMaxPstring)

HAB     hab;            /* Anchor-block handle */
PSZ     pszAppName;     /* Application name */
PSZ     pszKeyName;     /* Key name */
PSZ     pszDefault;     /* Default string */
PVOID   pProfileString; /* Profile string */
USHORT  usMaxPstring;   /* Maximum number */

USHORT  usPstring;      /* Actual number */
```

WinQueryProgramTitles

```
#define INCL_WINPROGRAMLIST   /* Or use INCL_WIN or INCL_PM */

BOOL          fSuccess = WinQueryProgramTitles (hab, hprogGroup, aprogeBuffer, usBufferLen,
                                                pusTotal)

HAB           hab;          /* Anchor-block handle */
HPROGRAM      hprogGroup;   /* Handle of the program group whose information is to be
                               returned */
PPROGRAMENTRY aprogeBuffer; /* Program information buffer */
USHORT        usBufferLen;  /* Buffer length */
PUSHORT       pusTotal;     /* Total number of structures */

BOOL          fSuccess;     /* Success indicator */
```

WinQueryQueueInfo

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL     fSuccess = WinQueryQueueInfo (hmq, pmqiMqinfo, cbCopied)

HMQ      hmq;        /* Queue handle */
PMQINFO  pmqiMqinfo; /* Message queue information structure to contain the queue information
                        */
USHORT   cbCopied;   /* Size of message queue information structure that is provided (in
                        bytes) */

BOOL     fSuccess;   /* Success indicator */
```

WinQueryQueueStatus

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

ULONG  flStatus = WinQueryQueueStatus (hwndDeskTop)

HWND   hwndDeskTop;  /* Desktop-window handle */

ULONG  flStatus;     /* Status information */
```

WinQuerySessionTitle

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT  usRetCode = WinQuerySessionTitle (hab, usSession, pszTitle, usTitlelen)

HAB     hab;        /* Anchor-block handle */
USHORT  usSession;  /* IBM Operating System/2 session identity of application whose title is
                       requested */
PSZ     pszTitle;   /* Task title */
USHORT  usTitlelen; /* Maximum length of data returnable in bytes */

USHORT  usRetCode;  /* Return code */
```

WinQuerySwitchEntry

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT    usRetCode = WinQuerySwitchEntry (hswitchSwitch, pswctlSwitchData)

HSWITCH   hswitchSwitch;    /* Handle to the task list entry */
PSWCNTRL  pswctlSwitchData; /* Switch control data */

USHORT    usRetCode;        /* Return code */
```

WinQuerySwitchHandle

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

HSWITCH  hswitchSwitch = WinQuerySwitchHandle (hwnd, idProcess)

HWND     hwnd;          /* Window handle of an application */
PID      idProcess;     /* Process identity of the application */

HSWITCH  hswitchSwitch; /* Switch list handle for the specified application */
```

WinQuerySwitchList

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT    usCount = WinQuerySwitchList (hab, pswblkSwitchEntries, usDataLength)

HAB       hab;                /* Anchor-block handle */
PSWBLOCK  pswblkSwitchEntries; /* Switch entries block */
USHORT    usDataLength;       /* Maximum length of data returnable in bytes */

USHORT    usCount;            /* Total number of switch list entries present in the system
                                 */
```

WinQuerySysColor

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

LONG   lRgbColor = WinQuerySysColor (hwndDeskTop, lColor, lReserved)

HWND   hwndDeskTop;  /* Desktop-window handle */
LONG   lColor;       /* System color-index value */
LONG   lReserved;    /* Reserved */

LONG   lRgbColor;    /* RGB value */
```

WinQuerySysModalWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND   hwndSysModal = WinQuerySysModalWindow (hwndDeskTop, fLock)

HWND   hwndDeskTop;  /* Desktop-window handle */
BOOL   fLock;        /* Window's lock-state indicator */

HWND   hwndSysModal; /* Handle of system modal window */
```

WinQuerySysPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

HPOINTER  hptrPointer = WinQuerySysPointer (hwndDeskTop, sIdentifier, fCopy)

HWND      hwndDeskTop;  /* Desktop-window handle */
SHORT     sIdentifier;  /* System-pointer identifier */
BOOL      fCopy;        /* Copy indicator */

HPOINTER  hptrPointer;  /* Pointer handle */
```

WinQuerySystemAtomTable

```
#define INCL_WINATOM   /* Or use INCL_WIN or INCL_PM */

HATOMTBL   hatomtblAtomTbl = WinQuerySystemAtomTable (VOID)

HATOMTBL   hatomtblAtomTbl;  /* System atom-table handle */
```

WinQuerySysValue

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

LONG   lValue = WinQuerySysValue (hwndDeskTop, sValueid)

HWND   hwndDeskTop;  /* Desktop-window handle */
SHORT  sValueid;     /* System-value identity */

LONG   lValue;       /* System value */
```

WinQueryTaskSizePos

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT  usRetCode = WinQueryTaskSizePos (hab, usScreenGroup, pswpPositionData)

HAB     hab;              /* Anchor-block handle */
USHORT  usScreenGroup;    /* Screen group */
PSWP    pswpPositionData; /* Window position and size data */

USHORT  usRetCode;        /* Return code */
```

WinQueryTaskTitle

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT   usRetCode = WinQueryTaskTitle (usSession, pszTitle, usTitlelen)

USHORT   usSession;  /* Session identity of application whose title is requested */
PSZ      pszTitle;   /* Task title */
USHORT   usTitlelen; /* Maximum length of data returnable in bytes */

USHORT   usRetCode;  /* Return code */
```

WinQueryUpdateRect

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinQueryUpdateRect (hwnd, prclPrc)

HWND    hwnd;     /* Handle of window whose update rectangle is to be queried */
PRECTL  prclPrc;  /* Update region that bounds the rectangle (in window coordinates) */

BOOL    fSuccess; /* Success indicator */
```

WinQueryUpdateRegion

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT  sComplexity = WinQueryUpdateRegion (hwnd, hrgn)

HWND   hwnd;        /* Handle of window whose update region is to be queried */
HRGN   hrgn;        /* Handle of the window's update region */

SHORT  sComplexity; /* Complexity of resulting region/error indicator */
```

WinQueryVersion

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

ULONG  flSysInf = WinQueryVersion (hab)

HAB    hab;      /* Anchor-block handle */

ULONG  flSysInf; /* System information within which the application is operating */
```

WinQueryWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND   hwndRelated = WinQueryWindow (hwnd, sCode, fLock)

HWND   hwnd;        /* Handle of window to query */
SHORT  sCode;       /* Type of window information */
BOOL   fLock;       /* Lock control */

HWND   hwndRelated; /* Window handle */
```

WinQueryWindowDC

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HDC    hdc = WinQueryWindowDC (hwnd)

HWND   hwnd;  /* Window handle */

HDC    hdc;   /* Device-context handle */
```

WinQueryWindowLockCount

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT  sLock = WinQueryWindowLockCount (hwnd)

HWND   hwnd;  /* Window handle */

SHORT  sLock; /* Window-lock count */
```

WinQueryWindowPos

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinQueryWindowPos (hwnd, pswp)

HWND   hwnd;     /* Window handle */
PSWP   pswp;     /* SWP structure */

BOOL   fSuccess; /* Success indicator */
```

WinQueryWindowProcess

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinQueryWindowProcess (hwnd, pidpid, pidtid)

HWND   hwnd;     /* Window handle */
PPID   pidpid;   /* Process identity of the thread that created the window */
PTID   pidtid;   /* Thread identity of the thread that created the window */

BOOL   fSuccess; /* Success indicator */
```

WinQueryWindowPtr

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

PVOID  pp = WinQueryWindowPtr (hwnd, sb)

HWND   hwnd;  /* Window handle */
SHORT  sb;    /* Index */

PVOID  pp;    /* Pointer value */
```

WinQueryWindowRect

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinQueryWindowRect (hwnd, prclRect)

HWND    hwnd;     /* Window handle */
PRECTL  prclRect; /* Window rectangle */

BOOL    fSuccess; /* Rectangle-returned indicator */
```

WinQueryWindowText

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT  sRetLen = WinQueryWindowText (hwnd, sLength, pchBuffer)

HWND   hwnd;      /* Window handle */
SHORT  sLength;   /* Length */
PCH    pchBuffer; /* Window text */

SHORT  sRetLen;   /* Length of returned text */
```

WinQueryWindowTextLength

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

SHORT  sRetLen = WinQueryWindowTextLength (hwnd)

HWND   hwnd;    /* Window handle */

SHORT  sRetLen; /* Length of the window text */
```

WinQueryWindowULong

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

ULONG  ulValue = WinQueryWindowULong (hwnd, sb)

HWND   hwnd;    /* Handle of window to be queried */
SHORT  sb;      /* Index */

ULONG  ulValue; /* Value contained in the window word */
```

WinQueryWindowUShort

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

USHORT  usValue = WinQueryWindowUShort (hwnd, sb)

HWND    hwnd;    /* Handle of window to be queried */
SHORT   sb;      /* Index */

USHORT  usValue; /* Value contained in the indicated window word */
```

WinReallocMem

```
#define INCL_WINHEAP   /* Or use INCL_WIN or INCL_PM */

NPBYTE  npbMem = WinReallocMem (hHeap, npbMemObj, usOld, usNew)

HHEAP   hHeap;     /* Handle to a heap */
NPBYTE  npbMemObj; /* Memory object */
USHORT  usOld;     /* Old size */
USHORT  usNew;     /* New size */

NPBYTE  npbMem;    /* Pointer to memory block */
```

WinRegisterClass

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fRegistered = WinRegisterClass (hab, pszClassName, pWndProc, flClassStyle, usExtra)

HAB     hab;          /* Anchor-block handle */
PSZ     pszClassName; /* Window-class name */
PFNWP   pWndProc;     /* Window-procedure identifier */
ULONG   flClassStyle; /* Default-window style */
USHORT  usExtra;      /* Reserved storage */

BOOL    fRegistered;  /* Window-class-registration indicator */
```

WinRegisterUserDatatype

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinRegisterUserDatatype (hab, sDatatype, sCount, asTypes)

HAB     hab;       /* Anchor-block handle */
SHORT   sDatatype; /* Data type code to be defined */
SHORT   sCount;    /* Number of elements */
PSHORT  asTypes;   /* Data type codes of structure components */

BOOL    fSuccess;  /* Success indicator */
```

WinRegisterUserMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinRegisterUserMsg (hab, usMsgid, sType1, sDir1, sType2, sDir2, sTyper)

HAB     hab;      /* Anchor-block handle */
USHORT  usMsgid;  /* Message identifier */
SHORT   sType1;   /* Datatype of message-parameter 1 */
SHORT   sDir1;    /* Direction of message-parameter 1 */
SHORT   sType2;   /* Datatype of message-parameter 2 */
SHORT   sDir2;    /* Direction of message-parameter 2 */
SHORT   sTyper;   /* Datatype of message reply */

BOOL    fSuccess; /* Success indicator */
```

WinRegisterWindowDestroy

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinRegisterWindowDestroy (hwnd, fRegister)

HWND   hwnd;      /* Window handle */
BOOL   fRegister; /* Registration indicator */

BOOL   fSuccess;  /* Success indicator */
```

WinReleaseHook

```
#define INCL_WINHOOKS   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinReleaseHook (hab, hmq, sHook, pAddress, Module)

HAB      hab;      /* Anchor-block handle */
HMQ      hmq;      /* Handle of message queue from which the hook is to be released */
SHORT    sHook;    /* Type of hook chain */
PFN      pAddress; /* Address of the hook routine */
HMODULE  Module;   /* Module handle */

BOOL     fSuccess; /* Success indicator */
```

WinReleasePS

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL  fSuccess = WinReleasePS (hps)

HPS   hps;      /* Handle of the cache presentation space to release */

BOOL  fSuccess; /* Success indicator */
```

WinRemovePresParam

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinRemovePresParam (hwnd, idAttrType)

HWND   hwnd;       /* Window handle */
ULONG  idAttrType; /* Attribute type identity */

BOOL   fSuccess;   /* Success indicator */
```

WinRemoveSwitchEntry

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

USHORT    usRetCode = WinRemoveSwitchEntry (hswitchSwitch)

HSWITCH   hswitchSwitch;  /* Switch-list (task-list) entry handle */

USHORT    usRetCode;      /* Success indicator */
```

WinScrollWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

SHORT   sComplexity = WinScrollWindow (hwnd, sDx, sDy, prclScroll, prclClip, hrgnUpdateRgn,
                                       prclUpdate, fsOptions)

HWND    hwnd;          /* Window handle */
SHORT   sDx;           /* Amount of horizontal scroll to the right (in device units) */
SHORT   sDy;           /* Amount of vertical scroll upward (in device units) */
PRECTL  prclScroll;    /* Scroll rectangle */
PRECTL  prclClip;      /* Clip rectangle */
HRGN    hrgnUpdateRgn; /* Update region */
PRECTL  prclUpdate;    /* Update rectangle */
USHORT  fsOptions;     /* Scroll options */

SHORT   sComplexity;   /* Complexity of resulting region/error indicator */
```

WinSendDlgItemMsg

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM */

MRESULT  mresReply = WinSendDlgItemMsg (hwndDlg, idItem, usMsgid, mpParam1, mpParam2)

HWND     hwndDlg;   /* Parent-window handle */
USHORT   idItem;    /* Identity of the child window */
USHORT   usMsgid;   /* Message identity */
MPARAM   mpParam1;  /* Message parameter 1 */
MPARAM   mpParam2;  /* Message parameter 2 */

MRESULT  mresReply; /* Message-return data */
```

WinSendMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

MRESULT  mresReply = WinSendMsg (hwnd, usMsgid, mpParam1, mpParam2)

HWND     hwnd;      /* Window handle */
USHORT   usMsgid;   /* Message identity */
MPARAM   mpParam1;  /* Parameter 1 */
MPARAM   mpParam2;  /* Parameter 2 */

MRESULT  mresReply; /* Message-return data */
```

WinSetAccelTable

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetAccelTable (hab, haccelAccel, hwndFrame)

HAB     hab;         /* Anchor-block handle */
HACCEL  haccelAccel; /* Accelerator-table handle */
HWND    hwndFrame;   /* Frame-window handle */

BOOL    fSuccess;    /* Success indicator */
```

WinSetActiveWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinSetActiveWindow (hwndDeskTop, hwnd)

HWND   hwndDeskTop; /* Desktop-window handle */
HWND   hwnd;        /* Window handle */

BOOL   fSuccess;    /* Active-window-set indicator */
```

WinSetCapture

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetCapture (hwndDesktop, hwnd)

HWND   hwndDesktop; /* Desktop-window handle, or HWND_DESKTOP */
HWND   hwnd;        /* Handle of the window that is to receive all pointing device messages
                    */

BOOL   fSuccess;    /* Success indicator */
```

WinSetClassMsgInterest

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetClassMsgInterest (hab, pszClassName, usMsgClass, sControl)

HAB     hab;          /* Anchor-block handle */
PSZ     pszClassName; /* Window-class name */
USHORT  usMsgClass;   /* Message class to have interest level set */
SHORT   sControl;     /* Interest identifier for the message class */

BOOL    fSuccess;     /* Interest-changed indicator */
```

WinSetClipbrdData

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL    fDataPlaced = WinSetClipbrdData (hab, ulh, usfmt, fsFmtInfo)

HAB     hab;         /* Anchor-block handle */
ULONG   ulh;         /* Handle */
USHORT  usfmt;       /* Format */
USHORT  fsFmtInfo;   /* Information */

BOOL    fDataPlaced; /* Data-placed indicator */
```

WinSetClipbrdOwner

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL  fSuccess = WinSetClipbrdOwner (hab, hwnd)

HAB   hab;      /* Anchor-block handle */
HWND  hwnd;     /* Window handle of the new clipboard owner */

BOOL  fSuccess; /* Success indicator */
```

WinSetClipbrdViewer

```
#define INCL_WINCLIPBOARD   /* Or use INCL_WIN or INCL_PM */

BOOL  fSuccess = WinSetClipbrdViewer (hab, hwndNewClipViewer)

HAB   hab;               /* Anchor-block handle */
HWND  hwndNewClipViewer; /* Window handle of the new clipboard viewer */

BOOL  fSuccess;          /* Success indicator */
```

WinSetCp

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetCp (hmq, usCodePage)

HMQ     hmq;        /* Message-queue handle */
USHORT  usCodePage; /* Code page */

BOOL    fSuccess;   /* Success indicator */
```

WinSetDlgItemShort

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinSetDlgItemShort (hwndDlg, idItem, usValue, fSigned)

HWND    hwndDlg;  /* Parent-window handle */
USHORT  idItem;   /* Identity of the child window whose text is to be changed */
USHORT  usValue;  /* Integer value used to generate the dialog item text */
BOOL    fSigned;  /* Sign indicator */

BOOL    fSuccess; /* Success indicator */
```

WinSetDlgItemText

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL    fSuccess = WinSetDlgItemText (hwndDlg, idItem, pszText)

HWND    hwndDlg;  /* Parent-window handle */
USHORT  idItem;   /* Identity of the child window whose text is to be set */
PSZ     pszText;  /* Source string */

BOOL    fSuccess; /* Success indicator */
```

WinSetErrorInfo

```
VOID       = WinSetErrorInfo (erridIdError, fsOptions, usargs)

ERRORID   erridIdError;  /* Error identity to be set */
USHORT    fsOptions;     /* Options */
USHORT    usargs;        /* Parameter words */
```

WinSetFocus

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinSetFocus (hwndDeskTop, hwndNewFocus)

HWND   hwndDeskTop;  /* Desktop-window handle */
HWND   hwndNewFocus; /* Window handle to receive the focus */

BOOL   fSuccess;     /* Success indicator */
```

WinSetHook

```
#define INCL_WINHOOKS   /* Or use INCL_WIN or INCL_PM */

BOOL     fSuccess = WinSetHook (hab, hmq, sHookType, pHookProc, Module)

HAB      hab;       /* Anchor-block handle */
HMQ      hmq;       /* Queue identity */
SHORT    sHookType; /* Hook chain type */
PFN      pHookProc; /* Address of the application hook procedure */
HMODULE  Module;    /* Resource identity */

BOOL     fSuccess;  /* Success indicator */
```

WinSetKeyboardStateTable

```
#define INCL_WININPUT   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetKeyboardStateTable (hwndDeskTop, abKeyStateTable, fSet)

HWND   hwndDeskTop;     /* Desktop-window handle */
PBYTE  abKeyStateTable; /* Key state table */
BOOL   fSet;            /* Set indicator */

BOOL   fSuccess;        /* Success indicator */
```

WinSetMsgInterest

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetMsgInterest (hwnd, usMsgClass, sControl)

HWND    hwnd;       /* Window handle */
USHORT  usMsgClass; /* Message class to have interest level set */
SHORT   sControl;   /* Interest-identifier for the message class */

BOOL    fSuccess;   /* Interest-changed indicator */
```

WinSetMsgMode

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinSetMsgMode (hab, pszClassName, sControl)

HAB    hab;          /* Anchor block handle */
PSZ    pszClassName; /* Window class name */
SHORT  sControl;     /* Message mode identifier */

BOOL   fSuccess;     /* Message delay indicator */
```

WinSetMultWindowPos

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetMultWindowPos (hab, aSwp, cCount)

HAB     hab;      /* Anchor-block handle */
PSWP    aSwp;     /* Array */
USHORT  cCount;   /* Window count */

BOOL    fSuccess; /* Positioning success indicator */
```

WinSetOwner

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetOwner (hwnd, hwndNewOwner)

HWND   hwnd;         /* Window handle whose owner window is to be changed */
HWND   hwndNewOwner; /* Handle of the new owner */

BOOL   fSuccess;     /* Success indicator */
```

WinSetParent

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetParent (hwnd, hwndNewParent, fRedraw)

HWND   hwnd;          /* Window handle */
HWND   hwndNewParent; /* New parent window handle */
BOOL   fRedraw;       /* Redraw indicator */

BOOL   fSuccess;      /* Parent-changed indicator */
```

WinSetPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL      fSuccess = WinSetPointer (hwndDeskTop, hptrNewPointer)

HWND      hwndDeskTop;    /* Desktop-window handle */
HPOINTER  hptrNewPointer; /* New pointer handle */

BOOL      fSuccess;       /* Pointer-updated indicator */
```

WinSetPointerPos

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetPointerPos (hwndDeskTop, sx, sy)

HWND   hwndDeskTop;  /* Desktop-window handle */
SHORT  sx;           /* x position of pointer in screen coordinates */
SHORT  sy;           /* y position of pointer in screen coordinates */

BOOL   fSuccess;     /* Pointer position updated indicator */
```

WinSetPresParam

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetPresParam (hwnd, idAttrType, cbAttrValueLen, pAttrValue)

HWND   hwnd;           /* Window handle */
ULONG  idAttrType;     /* Attribute type identity */
ULONG  cbAttrValueLen; /* Byte count of the data passed in the AttrValue parameter */
PVOID  pAttrValue;     /* Attribute value */

BOOL   fSuccess;       /* Success indicator */
```

WinSetRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetRect (hab, prclrect, sLeft, sBottom, sRight, sTop)

HAB     hab;      /* Anchor-block handle */
PRECTL  prclrect; /* Rectangle to be updated */
SHORT   sLeft;    /* Left edge of rectangle */
SHORT   sBottom;  /* Bottom edge of rectangle */
SHORT   sRight;   /* Right edge of rectangle */
SHORT   sTop;     /* Top edge of rectangle */

BOOL    fSuccess; /* Success indicator */
```

WinSetRectEmpty

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetRectEmpty (hab, prclrect)

HAB     hab;      /* Anchor-block handle */
PRECTL  prclrect; /* Rectangle to be set empty */

BOOL    fSuccess; /* Success indicator */
```

WinSetSynchroMode

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinSetSynchroMode (hab, sMode)

HAB    hab;      /* Anchor-block handle */
SHORT  sMode;    /* Synchronization mode */

BOOL   fSuccess; /* Success indicator */
```

WinSetSysColors

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetSysColors (hwndDeskTop, flOptions, ulFormat, lStart, ulTablen,
                                   alTable)

HWND   hwndDeskTop;  /* Desktop-window handle */
ULONG  flOptions;    /* Options */
ULONG  ulFormat;     /* Format of entries in the table, as follows */
LONG   lStart;       /* Starting system color index */
ULONG  ulTablen;     /* Number of elements */
PLONG  alTable;      /* Table */

BOOL   fSuccess;     /* Success indicator */
```

WinSetSysModalWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetSysModalWindow (hwndDesktop, hwnd)

HWND   hwndDesktop;  /* Desktop-window handle, or HWND_DESKTOP */
HWND   hwnd;         /* Handle of window to become system-modal window */

BOOL   fSuccess;     /* Success indicator */
```

WinSetSysValue

```
#define INCL_WINSYS   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetSysValue (hwndDeskTop, sValueid, lValue)

HWND   hwndDeskTop;  /* Desktop-window handle */
SHORT  sValueid;     /* System-value identity */
LONG   lValue;       /* System value */

BOOL   fSuccess;     /* Value-set indicator */
```

WinSetWindowBits

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetWindowBits (hwnd, sb, flData, flMask)

HWND   hwnd;     /* Window handle */
SHORT  sb;       /* Zero-based index of the value to be set */
ULONG  flData;   /* Bit data to store in the window words */
ULONG  flMask;   /* Bits to be written indicator */

BOOL   fSuccess; /* Success indicator */
```

WinSetWindowPos

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetWindowPos (hwnd, hwndBehind, sx, sy, scx, scy, fsOptions)

HWND    hwnd;       /* Window handle */
HWND    hwndBehind; /* Relative window-placement order */
SHORT   sx;         /* Window position, x coordinate */
SHORT   sy;         /* Window position, y coordinate */
SHORT   scx;        /* Window size */
SHORT   scy;        /* Window size */
USHORT  fsOptions;  /* Window-positioning options */

BOOL    fSuccess;   /* Repositioning indicator */
```

WinSetWindowPtr

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetWindowPtr (hwnd, sb, pp)

HWND   hwnd;     /* Window handle */
SHORT  sb;       /* Zero-based index into the window words */
PVOID  pp;       /* Pointer value to store in the window words */

BOOL   fSuccess; /* Success indicator */
```

WinSetWindowText

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fResult = WinSetWindowText (hwnd, pszString)

HWND   hwnd;      /* Window handle */
PSZ    pszString; /* Window text */

BOOL   fResult;   /* Text-updated indicator */
```

WinSetWindowULong

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinSetWindowULong (hwnd, sb, ulData)

HWND   hwnd;      /* Window handle */
SHORT  sb;        /* Zero-based index of the value to be set */
ULONG  ulData;    /* Unsigned, long integer value to store in the window words */

BOOL   fSuccess; /* Success indicator */
```

WinSetWindowUShort

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinSetWindowUShort (hwnd, sb, usData)

HWND    hwnd;      /* Window handle */
SHORT   sb;        /* Zero-based index of the value to be set */
USHORT  usData;    /* Unsigned, short integer value to store in the window words */

BOOL    fSuccess; /* Success indicator */
```

WinShowCursor

```
#define INCL_WINCURSORS   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinShowCursor (hwnd, fShow)

HWND   hwnd;      /* Handle of window to which the cursor belongs */
BOOL   fShow;     /* Show indicator */

BOOL   fSuccess; /* Success indicator */
```

WinShowPointer

```
#define INCL_WINPOINTERS   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinShowPointer (hwndDeskTop, fShow)

HWND   hwndDeskTop;  /* Desktop-window handle */
BOOL   fShow;        /* Level-update indicator */

BOOL   fSuccess;     /* Display-level-updated indicator */
```

WinShowTrackRect

```
#define INCL_WINTRACKRECT   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinShowTrackRect (hwnd, fShow)

HWND   hwnd;      /* Window handle */
BOOL   fShow;     /* Show indicator */

BOOL   fSuccess; /* Success indicator */
```

WinShowWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL   fSuccess = WinShowWindow (hwnd, fNewVisibility)

HWND   hwnd;           /* Window handle */
BOOL   fNewVisibility; /* New visibility state */

BOOL   fSuccess;       /* Visibility changed indicator */
```

WinStartTimer

```
#define INCL_WINTIMER   /* Or use INCL_WIN or INCL_PM */

USHORT  usRet = WinStartTimer (hab, hwnd, idTimer, usTimeout)

HAB     hab;       /* Anchor-block handle */
HWND    hwnd;      /* Window handle that is part of the timer identification */
USHORT  idTimer;   /* Timer identifier */
USHORT  usTimeout; /* Delay time in milliseconds */

USHORT  usRet;     /* Return code */
```

WinStopTimer

```
#define INCL_WINTIMER   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinStopTimer (hab, hwnd, usTimer)

HAB     hab;      /* Anchor-block handle */
HWND    hwnd;     /* Window handle */
USHORT  usTimer;  /* Timer identifier */

BOOL    fSuccess; /* Success indicator */
```

WinSubclassWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

PFNWP  pOldWindowProc = WinSubclassWindow (hwnd, pNewWindowProc)

HWND   hwnd;           /* Handle of window that is being subclassed */
PFNWP  pNewWindowProc; /* New window procedure */

PFNWP  pOldWindowProc; /* Old window procedure */
```

WinSubstituteStrings

```
#define INCL_WINDIALOGS   /* Or use INCL_WIN or INCL_PM */

SHORT  sDestRet = WinSubstituteStrings (hwnd, pszSrc, sDestMax, pszDest)

HWND   hwnd;     /* Handle of window that processes the call */
PSZ    pszSrc;   /* Source string */
SHORT  sDestMax; /* Maximum number of characters returnable */
PSZ    pszDest;  /* Resultant string */

SHORT  sDestRet; /* Actual number of characters returned */
```

WinSubtractRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fNonempty = WinSubtractRect (hab, prclDest, prclSrc1, prclSrc2)

HAB     hab;       /* Anchor-block handle */
PRECTL  prclDest;  /* Result */
PRECTL  prclSrc1;  /* First source rectangle */
PRECTL  prclSrc2;  /* Second source rectangle */

BOOL    fNonempty; /* Not-empty indicator */
```

WinSwitchToProgram

```
#define INCL_WINSWITCHLIST   /* Or use INCL_WIN or INCL_PM */

USHORT    usRetCode = WinSwitchToProgram (hswitchSwHandle)

HSWITCH   hswitchSwHandle;  /* Switch list entry handle of program to be activated */

USHORT    usRetCode;        /* Return code */
```

WinTerminate

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM. Also in COMMON section */

BOOL  fTerminated = WinTerminate (hab)

HAB   hab;         /* Anchor-block handle */

BOOL  fTerminated; /* Termination indicator */
```

WinTerminateApp

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinTerminateApp (happ)

HAPP   happ;     /* Application handle */

BOOL   fSuccess; /* Success indicator */
```

WinThrow

```
#define INCL_WINCATCHTHROW   /* Or use INCL_WIN or INCL_PM */

VOID         = WinThrow (pctchbfCatchBuf, sThrowBack)

PCATCHBUF   pctchbfCatchBuf;  /* Saved execution-environment buffer */
SHORT       sThrowBack;       /* Return value */
```

WinTrackRect

```
#define INCL_WINTRACKRECT   /* Or use INCL_WIN or INCL_PM */

BOOL        fSuccess = WinTrackRect (hwnd, hps, ptiTrackinfo)

HWND        hwnd;         /* Window handle where tracking is to take place */
HPS         hps;          /* Presentation-space handle */
PTRACKINFO  ptiTrackinfo; /* Track information */

BOOL        fSuccess;     /* Success indicator */
```

WinTranslateAccel

```
#define INCL_WINACCELERATORS   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinTranslateAccel (hab, hwnd, haccelAccel, pQmsg)

HAB     hab;         /* Anchor-block handle */
HWND    hwnd;        /* Destination window */
HACCEL  haccelAccel; /* Accelerator-table handle */
PQMSG   pQmsg;       /* Message to be translated */

BOOL    fSuccess;    /* Success indicator */
```

WinUnionRect

```
#define INCL_WINRECTANGLES   /* Or use INCL_WIN or INCL_PM */

BOOL    fNonempty = WinUnionRect (hab, prclDest, prclSrc1, prclSrc2)

HAB     hab;       /* Anchor-block handle */
PRECTL  prclDest;  /* Bounding rectangle */
PRECTL  prclSrc1;  /* First source rectangle */
PRECTL  prclSrc2;  /* Second source rectangle */

BOOL    fNonempty; /* Non-empty indicator */
```

WinUpdateWindow

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinUpdateWindow (hwnd)

HWND   hwnd;     /* Window handle */

BOOL   fSuccess; /* Window-updated indicator */
```

WinUpper

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

USHORT  usRetLen = WinUpper (hab, usCodepage, usCountry, pszString)

HAB     hab;        /* Anchor-block handle */
USHORT  usCodepage; /* Code page */
USHORT  usCountry;  /* Country code */
PSZ     pszString;  /* String to be converted to uppercase */

USHORT  usRetLen;   /* Length of converted string */
```

WinUpperChar

```
#define INCL_WINCOUNTRY   /* Or use INCL_WIN or INCL_PM */

USHORT  usOutchar = WinUpperChar (hab, usCodepage, usCountry, usInchar)

HAB     hab;        /* Anchor-block handle */
USHORT  usCodepage; /* Code page */
USHORT  usCountry;  /* Country code */
USHORT  usInchar;   /* Character to be translated to uppercase */

USHORT  usOutchar;  /* Translated character */
```

WinValidateRect

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinValidateRect (hwnd, prclRect, fIncludeClippedChildren)

HWND    hwnd;                   /* Handle of window whose update region is changed */
PRECTL  prclRect;               /* Rectangle to be subtracted from the window's update
                                   region */
BOOL    fIncludeClippedChildren; /* Validation-scope indicator */

BOOL    fSuccess;               /* Success indicator */
```

WinValidateRegion

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

BOOL   fSuccess = WinValidateRegion (hwnd, hrgn, fIncludeClippedChildren)

HWND   hwnd;                   /* Handle of window whose update region is changed */
HRGN   hrgn;                   /* Handle of subtracted region */
BOOL   fIncludeClippedChildren; /* Validation-scope indicator */

BOOL   fSuccess;               /* Success indicator */
```

WinWaitMsg

```
#define INCL_WINMESSAGEMGR   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinWaitMsg (hab, usFirst, usLast)

HAB     hab;      /* Anchor-block handle */
USHORT  usFirst;  /* First message identity */
USHORT  usLast;   /* Last message identity */

BOOL    fSuccess; /* Success indicator */
```

WinWindowFromDC

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND  hwnd = WinWindowFromDC (hdc)

HDC   hdc;  /* Device-context handle */

HWND  hwnd; /* Window handle */
```

WinWindowFromID

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND    hwnd = WinWindowFromID (hwndParent, usidentifier)

HWND    hwndParent;   /* Parent-window handle */
USHORT  usidentifier; /* Identity of the child window */

HWND    hwnd;         /* Window handle */
```

WinWindowFromPoint

```
#define INCL_WINWINDOWMGR   /* Or use INCL_WIN or INCL_PM */

HWND     hwndFound = WinWindowFromPoint (hwndParent, pptlPoint, fEnumChildren, fLock)

HWND     hwndParent;    /* Window handle whose child windows are to be tested */
PPOINTL  pptlPoint;     /* The point to be tested */
BOOL     fEnumChildren; /* Test control */
BOOL     fLock;         /* Lock control */

HWND     hwndFound;     /* Window handle beneath Point */
```

WinWriteProfileData

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL    fSuccess = WinWriteProfileData (hab, pszAppName, pszKeyName, pValue, usSize)

HAB     hab;        /* Anchor-block handle */
PSZ     pszAppName; /* Text string */
PSZ     pszKeyName; /* Text string */
PVOID   pValue;     /* Value data */
USHORT  usSize;     /* Size of value data */

BOOL    fSuccess;   /* Success indicator */
```

WinWriteProfileString

```
#define INCL_WINSHELLDATA   /* Or use INCL_WIN or INCL_PM */

BOOL  fSuccess = WinWriteProfileString (hab, pszAppName, pszKeyName, pszValue)

HAB   hab;        /* Anchor block handle */
PSZ   pszAppName; /* Text string */
PSZ   pszKeyName; /* Text string */
PSZ   pszValue;   /* Text string */

BOOL  fSuccess;   /* Success indicator */
```

Chapter 10. Functions Supplied by Applications

DialogProc

```
MRESULT  mresReply = DialogProc (hwnd, usmsg, mpParam1, mpParam2)

HWND     hwnd;      /* Handle of the window to which the message applies */
USHORT   usmsg;     /* Message identity */
MPARAM   mpParam1;  /* Message parameter 1 */
MPARAM   mpParam2;  /* Message parameter 2 */

MRESULT  mresReply; /* Message-return data */
```

WndProc

```
MRESULT  mresReply = WndProc (hwnd, usmsg, mpParam1, mpParam2)

HWND     hwnd;      /* Window handle */
USHORT   usmsg;     /* Message identity */
MPARAM   mpParam1;  /* Message parameter 1 */
MPARAM   mpParam2;  /* Message parameter 2 */

MRESULT  mresReply; /* Message-return data */
```

CodePageChangeHook

```
VOID     return = CodePageChangeHook (hmq, usOldCodePage, usNewCodePage)

HMQ      hmq;           /* Message-queue handle */
USHORT   usOldCodePage; /* Previous code page */
USHORT   usNewCodePage; /* New code page */
```

HelpHook

```
BOOL     f = HelpHook (hab, sMode, sTopic, sSubTopic, prclPosition)

HAB      hab;          /* Anchor-block handle */
SHORT    sMode;        /* Help mode */
SHORT    sTopic;       /* Topic identifier */
SHORT    sSubTopic;    /* Subtopic identifier */
PRECTL   prclPosition; /* Rectangle */

BOOL     f;            /* Indicator as to whether next hook in the chain is called */
```

InputHook

```
BOOL     fProcessed = InputHook (hab, pQmsg, fsOptions)

HAB      hab;        /* Anchor-block handle */
PQMSG    pQmsg;      /* A QMSG data structure */
USHORT   fsOptions;  /* Message removal options */

BOOL     fProcessed; /* Processed indicator */
```

JournalPlaybackHook

```
LONG   lTime = JournalPlaybackHook (hab, pqmsg, fSkip)

HAB    hab;   /* Anchor-block handle */
PQMSG  pqmsg; /* Data structure where the message to be played back is returned */
BOOL   fSkip; /* Indicator as to whether the next message should be played back */

LONG   lTime; /* Waiting time */
```

JournalRecordHook

```
BOOL   fSuccess = JournalRecordHook (hab, pqmsg)

HAB    hab;      /* Anchor-block handle */
PQMSG  pqmsg;    /* Data structure that contains the message to be recorded */

BOOL   fSuccess; /* The return value from this hook is ignored */
```

LoaderHook

```
BOOL   fProcessed = LoaderHook (hab, sContext, pszlibname, phlibLibhandle, pszprocname,
                                pwndproc, pfSuccess)

HAB    hab;            /* Anchor-block handle */
SHORT  sContext;       /* Origin of call to hook */
PSZ    pszlibname;     /* Library name */
PHLIB  phlibLibhandle; /* Library handle */
PSZ    pszprocname;    /* Procedure name */
PFN    pwndproc;       /* Window procedure identifier */
PBOOL  pfSuccess;      /* Success indicator */

BOOL   fProcessed;     /* Processing indicator */
```

MsgCtlHook

```
BOOL   fProcessed = MsgCtlHook (hab, sContext, hwnd, pszClassName, usMsgClass, sControl,
                                pfSuccess)

HAB    hab;          /* Anchor-block handle */
SHORT  sContext;     /* Origin of call to hook */
HWND   hwnd;         /* Window handle */
PSZ    pszClassName; /* Window class name */
USHORT usMsgClass;   /* Message class */
SHORT  sControl;     /* Control setting */
PBOOL  pfSuccess;    /* Success indicator */

BOOL   fProcessed;   /* Processing indicator */
```

MsgFilterHook

```
BOOL   fProcessed = MsgFilterHook (hab, pQmsg, usContext)

HAB    hab;       /* Anchor-block handle */
PQMSG  pQmsg;     /* A queue message data structure */
USHORT usContext; /* Context in which the hook has been called */

BOOL   fProcessed; /* Processed indicator */
```

RegisterUserMsg

```
BOOL   fProcessed = RegisterUserMsg (hab, sContext, usMsgid, sType1, sDir1, sType2, sDir2,
                                     sTyper, sCount, asTypes, pfSuccess)

HAB    hab;       /* Anchor-block handle */
SHORT  sContext;  /* Origin of call to hook */
USHORT usMsgid;   /* Message identifier */
SHORT  sType1;    /* Data type of message-parameter 1 */
SHORT  sDir1;     /* Direction of message-parameter 1 */
SHORT  sType2;    /* Data type of message-parameter 2 */
SHORT  sDir2;     /* Direction of message-parameter 2 */
SHORT  sTyper;    /* Data type of message reply */
SHORT  sCount;    /* Number of elements */
PSHORT asTypes;   /* Data types of structure components */
PBOOL  pfSuccess; /* Success indicator */

BOOL   fProcessed; /* Processing indicator */
```

SendMsgHook

```
VOID        return = SendMsgHook (hab, psmhssmh, fInterTask)

HAB         hab;        /* Anchor-block handle */
PSMHSTRUCT  psmhssmh;   /* Send message hook structure */
BOOL        fInterTask; /* Intertask indicator */
```

Index

H

I

J

K

T

U

V

W

X

IBM United Kingdom
International Products Limited
PO Box 41, North Harbour
Portsmouth, PO6 3AU
England

Printed in Denmark by Rosendahl · Esbjerg